

Technical Report

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

ISRN L3COM/MARITIMESYSTEMS/TR -- 2011/001

Contract Number: N00014-09-C-0618

Prepared For:



Prepared by:



Maritime Systems
9 Malcolm Hoyt Drive
Newburyport, MA 01950 U.S.A.

Notice: This material may be protected by copyright law (Title 17 U.S. Code).



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

This page intentionally left blank.



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

REPORT DOCUMENTATION PAGE		<i>Form Approved OMB No. 0704-0188</i>
<small>Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</small>		
1. REPORT DATE (DD-MM-YYYY) 31-03-2011	2. REPORT TYPE Research	3. DATES COVERED (From - To)
4. TITLE AND SUBTITLE Intelligent Advanced Communications Assured Services Session Initiated Protocol Telephony Feasibility for the US Navy, General Implementation		5a. CONTRACT NUMBER N00014-09-C-0618
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Binns, Todd D; Principle Investigator Oram, Jerald; Investigator Keohane, Stephen; Investigator		5d. PROJECT NUMBER N.A.
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) L-3 Maritime Systems 9 Malcolm Hoyt Drive Newburyport, MA 01950-4017		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research 875 N Randolph St., Suite 1425 Arlington, VA 22203-1995		10. SPONSOR/MONITOR'S ACRONYM(S) ONR
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release, distribution is unlimited		
13. SUPPLEMENTARY NOTES		



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

14. ABSTRACT

The objective of this research paper is to detail our exploration of open source SIP solutions as a viable alternative to building an Assured Services SIP end instrument (aka IP phone) as defined by the DOD Unified Capabilities and requirements specification.

This report is based on the following research and validation testing:

- L-3 Maritime Systems internal research,
- L-3 Maritime Systems VoIP lab research,
- L-3 Maritime Systems prototyping of a VoIP Communications Terminal,
- Unified Capabilities Requirements 2008 Change 1,
- L-3 Maritime Systems Technical Report *Intelligent Advanced Communications IP Telephony Feasibility for the US Navy*, Volumes 1 and 2 (ISRN L3COM/Maritime Systems/TR-2007/001),
- L-3 Maritime Systems Technical Report *Intelligent Advanced Communications IP Telephony Feasibility for the US Navy*, Phase 2 (ISRN L3COM/Maritime Systems/TR-2009/001),
- OSIP, an open source SIP stack,
- SipXtapi, an open source SIP stack,
- SipXtapi-3.3.0, an open source SIP stack,
- REDCOM Slice2100 soft switch with Transip, HDX,
- GL Communications VoIP quality test tool,
- Wireshark,
- Licensing requirements and legal implications of various open source VoIP solutions.

The major findings are:

- SIP stack behavior and characteristics varies substantially among vendors.
- Unlike traditional TDM based phones, SIP stacks and consequently SIP phones are somewhat switch specific in terms of compatibility and vary substantially in terms of what they chose to support. Though one can plug a TDM-based phone into another TDM network and base functionality will work, the same is not true for IP phones.
- 2 of the 3 open source SIP stacks we evaluated were maintained outside the United States.
- The SipxTapi package in particular has over 13,000 source files spread out over 2,500 folders.

Vetting the source for potential malware and or back doors or any type of security risk would be extremely difficult

- None of the open source stacks we surveyed had support for IPV6 fully integrated with the SIP stack or at all.
- IP based voice and video networks are already in wide use today in networks where there is little or no control over competing network traffic. A closed network environment on board a ship would provide an optimal environment for IP based communications.
- A stack based on OSIP would likely be the best candidate for developing a UCR defined SIP end instrument for the Navy or any DOD entity, especially for host based solutions.



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

15. SUBJECT TERMS VoIP, SIP, H.323, US Navy, Vessels, Communications, Network, SVoIP, VoSIP, SVoSIP, AS-SIP, Assured Services SIP					
16. SECURITY CLASSIFICATION OF: U			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON Christopher P. Rigano (CISSP)
a. REPORT	b. ABSTRACT	c. THIS PAGE 2	UU		19b. TELEPHONE NUMBER (include area code) 703.696.65942

Standard Form 298 (Rev.
8-98)
Prescribed by ANSI Std. Z39.18



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

This page intentionally left blank.



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Table of Contents

<i>ABSTRACT.....</i>	<i>XIII</i>
<i>CHAPTER 1 EXECUTIVE SUMMARY.....</i>	<i>1</i>
1.1 OBJECTIVES.....	1
1.2 INTRODUCTION.....	1
<i>CHAPTER 2 VOIP REVIEW.....</i>	<i>5</i>
2.1 OVERVIEW.....	5
2.2 VOIP USAGE.....	5
2.3 VOIP PROTOCOL SUITES.....	6
2.4 SESSION INITIATION PROTOCOL.....	7
2.5 SIP LIMITATIONS.....	11
2.5.3 Selected SIP Features Summary.....	18
2.6 SESSION INITIATION PROTOCOL (SIP) FEASIBILITY.....	18
2.6.1 Introduction to SIP.....	18
2.6.2 What is SIP?.....	19
2.6.3 Core SIP Specifications.....	20
2.6.4 SIP Infrastructure Extensions (General Uses).....	22
2.6.5 SIP Limitations.....	23
2.6.6 Basic Flows.....	23
2.6.7 Basic SIP Call.....	26
2.6.8 Implementation of Selected Features.....	28
2.7 SUMMARY.....	43
<i>CHAPTER 3 IAC SECURITY.....</i>	<i>45</i>
3.1. INTRODUCTION.....	45
3.2. SECURITY.....	45
3.2.1 Security Threats.....	45
3.2.3 Security Tactics.....	46
3.2.4 Software Management.....	47
3.2.5 Security Policies.....	47
3.3. GOVERNMENT AND INDUSTRY CONTRIBUTIONS.....	48
3.3.1 Defense Information Systems Agency.....	48
3.3.2 National Institute of Standards and Technology.....	48
3.3.3 Other Relevant Contributors.....	49
<i>CHAPTER 4 OPEN SOURCE FEASIBILITY.....</i>	<i>55</i>
4.1 INTRODUCTION.....	55
4.2 SIP AND ASSURED SERVICES SIP.....	56
4.2.1 AS-SIP Messages.....	56



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Table of Contents - Continued

4.2.3 AS-SIP Multilevel Precedence and Preemption (MLPP)	60
4.2.4 AS-SIP Precedence Rules	62
4.3 OPEN SOURCE	63
4.3.1 Open Source SIP Stack Evaluations	64
4.4 BUILDING SIPXTAPI	77
4.4.1 Active SipXtapi Control	89
4.4.2 Marshalling Layer	93
4.5 CONCLUSION	94
CHAPTER 5 UNICOI AND EMBEDDED	96
5.1 INTRODUCTION	97
5.2 OVERVIEW	98
5.3 INSTAVOIP FAMILY	98
5.3.2 InstaVoIP Mobile	98
5.3.3 InstaVoIP Module	99
5.3.4 Example Applications	99
5.4 FEATURES OVERVIEW	101
5.5 ARCHITECTURE	102
5.5.1 RTOS/OS, Drivers, Networking Stack, and File System	103
5.5.2 Fusion Common Library (FCL) and Porting Layer	103
5.5.3 Audio Driver Channel	104
5.5.4 VoIP Networking Protocols	104
5.5.5 Voice Engine Audio Processing	105
5.5.6 Call Manager	105
5.5.7 Info Subsystem	106
5.5.8 Configuration Web Application and Web Service	106
5.5.9 Call Manager Web Service	106
5.5.10 Expansion APIs	107
5.5.11 Call Manager Interface	107
5.5.12 Info Subsystem	107
5.5.13 Configuration Web Application	107
5.5.14 Voice Engine	108
5.6 PORTING	108
5.6.1 FCL	108
5.6.2 Audio Driver Channel	108
5.7 Info Subsystem	109
CHAPTER 6 UNICOI GRAPHICAL USER INTERFACE	111
6.1 WINDOWS GUI	111
6.1.1 Introduction	111
6.1.2 Cougar Win32 Net Client	111



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Table of Contents - Continued

6.1.3	<i>Features</i>	111
6.1.4	<i>Compilation</i>	112
6.2	<i>CONFIGURATION</i>	113
6.2.1	<i>Provisioning Display</i>	114
6.3	<i>MAKING A SIP CALL USING COUGAR WIN32 DEMO</i>	125
6.3.1	<i>Active 2 Party SIP Call</i>	126
6.3.2	<i>Summary</i>	127
6.4	<i>HEADLESS CLIENT</i>	128
6.4.1	<i>Features</i>	128
6.4.2	<i>Compilation</i>	128
6.4.3	<i>Configuration</i>	129
6.4.4	<i>Execution</i>	130
6.4.5	<i>Media</i>	132
6.4.6	<i>HttpCmd Utility</i>	132
6.5	<i>PORTAUDIO LIBRARY</i>	135
6.5.1	<i>License</i>	136
6.5.2	<i>Overview of PortAudio</i>	137
6.6	<i>SUMMARY</i>	138
CHAPTER 7 NETWORK VOIP PERFORMANCE		139
7.1	<i>INTRODUCTION</i>	139
7.2	<i>VOICE QUALITY TESTING</i>	139
7.2.1	<i>Voice Quality Testing Overview</i>	140
7.3	<i>VOICE QUALITY METRICS</i>	143
7.3.1	<i>One Way Delay</i>	143
7.3.3	<i>Mean Opinion Score</i>	143
7.3.4	<i>PAMS_LE, PAMS_LQ</i>	143
7.3.5	<i>PSQM and PSQM+</i>	143
7.3.6	<i>MOS</i>	146
7.4	<i>PHASES OF TESTING</i>	146
7.4.5	<i>Software Setup</i>	150
7.4.6	<i>VQT Setup</i>	167
7.4.7	<i>Transfer/Test Monitoring</i>	177
7.4.8	<i>Stage III: End-to-End Communication Verification</i>	192
7.4.9	<i>Detailed Implementation</i>	192
7.5	<i>SUMMARY</i>	203
CHAPTER 8 CONCLUSION		205
8.1	<i>INTRODUCTION</i>	205
8.2	<i>CONCLUSION</i>	205



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Table of Contents - Continued

<i>APPENDIX A GNU GENERAL PUBLIC LICENSE</i>	<i>207</i>
<i>APPENDIX B GNU LESSER GENERAL PUBLIC LICENSE.....</i>	<i>219</i>
<i>APPENDIX C TEST PLAN.....</i>	<i>223</i>
<i>APPENDIX D END-TO-END TEST COMMUNICATION VERIFICATION.....</i>	<i>233</i>
<i>D-1 END-TO-END COMMUNICATION VERIFICATION</i>	<i>235</i>
<i>D-2 DETAILED IMPLEMENTATION.....</i>	<i>235</i>
<i>D-3 HARDWARE SETUP</i>	<i>235</i>
<i>D-3.1 Data Path.....</i>	<i>237</i>
<i>D-4 SOFTWARE SETUP</i>	<i>237</i>
<i>D-4.1 VQuad Setup</i>	<i>237</i>
<i>D-4.2 Device Configuration.....</i>	<i>238</i>
<i>D-4.3 Modify or Create a VQuad Device Configuration for End-to-End VoIP Testing.....</i>	<i>239</i>
<i>D-4.4 Auto-Traffic Configuration</i>	<i>241</i>
<i>D-4.6 Modify or Create a VQuad Auto Traffic Configuration for End-to-End Testing.....</i>	<i>246</i>
<i>D-5 VQT SETUP</i>	<i>248</i>
<i>D-6 CHECKING VQT AUTO-MEASUREMENT SETUP.....</i>	<i>254</i>
<i>D-7 TEST EXECUTION</i>	<i>256</i>
<i>D-7.1 Preparation.....</i>	<i>256</i>
<i>D-7.2 Start VQT.....</i>	<i>257</i>
<i>D-7.3 Start VQuad</i>	<i>257</i>
<i>D-7.4 Transfer/Test Monitoring.....</i>	<i>258</i>
<i>D-7.5 VQT Testing Status</i>	<i>260</i>
<i>D-7.6 File Movement Monitoring (Windows).....</i>	<i>261</i>
<i>D-8 TEST RESULTS.....</i>	<i>263</i>
<i>D-8.1 PAMS</i>	<i>264</i>
<i>D-8.2 PSQM and PSQM+</i>	<i>264</i>
<i>D-8.3 PESQ.....</i>	<i>265</i>
<i>D-8.4 TEST FILE VERIFICATION/AUDITING</i>	<i>265</i>
<i>APPENDIX E LOOPBACK TESTING AND VERIFICATION (BASELINE VERIFICATION).....</i>	<i>267</i>
<i>E-1. OVERVIEW AND ASSUMPTIONS</i>	<i>269</i>
<i>E-2. OVERVIEW</i>	<i>269</i>
<i>E-3. ASSUMPTIONS</i>	<i>269</i>
<i>E-4. BASELINE VERIFICATION (LOOPBACK TESTING)</i>	<i>270</i>
<i>E-4.1 Detailed Implementation.....</i>	<i>270</i>
<i>E.4.1.1 Test File Verification/Auditing</i>	<i>270</i>



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Table of Contents - Continued

<i>E-4.1.2</i>	<i>Hardware Setup</i>	<i>270</i>
<i>E-4.1.3</i>	<i>Single-PC/Single-UTA Loopback Test Setup.....</i>	<i>271</i>
<i>E-4.1.4</i>	<i>Data Path.....</i>	<i>271</i>
<i>E-4.1.6</i>	<i>VQuad Configuration.....</i>	<i>271</i>
<i>E-4.1.7</i>	<i>Modify or Create a VQuad Device Configuration for Loopback Testing</i>	<i>274</i>
<i>E-4.1.8</i>	<i>Auto-Traffic Configuration</i>	<i>276</i>
<i>E-4.1.9</i>	<i>Configure Device 1</i>	<i>277</i>
<i>E-4.1.10</i>	<i>Configure Device 2</i>	<i>282</i>
<i>E-4.3.11</i>	<i>Modify or Create a VQuad Auto Traffic Configuration</i>	<i>285</i>
<i>E-4.3.12</i>	<i>General Tab Fields</i>	<i>286</i>
<i>E-4.3.13</i>	<i>Tx Provisioning Tab Fields.....</i>	<i>287</i>
<i>E-4.3.14</i>	<i>Rx Provisioning Tab Fields</i>	<i>287</i>
<i>E-4.3.15</i>	<i>VQT Setup.....</i>	<i>287</i>
<i>E-4.3.16</i>	<i>Checking VQT Auto-Measurement Setup</i>	<i>293</i>
<i>E-5</i>	<i>TEST EXECUTION</i>	<i>295</i>
<i>E-5.1</i>	<i>Start VQT.....</i>	<i>296</i>
<i>E-5.2</i>	<i>Start VQuad</i>	<i>296</i>
<i>E-6</i>	<i>TRANSFER/TEST MONITORING</i>	<i>297</i>
<i>E-6.1</i>	<i>VQuad Status Monitoring</i>	<i>297</i>
<i>E-6.2</i>	<i>VQT Testing Monitoring.....</i>	<i>298</i>
<i>E-7</i>	<i>TEST RESULTS.....</i>	<i>302</i>
<i>E-7.1</i>	<i>Detailed Data.....</i>	<i>303</i>
<i>E-7.2</i>	<i>PAMS</i>	<i>303</i>
<i>E-7.3</i>	<i>PSQM and PSQM+</i>	<i>304</i>
<i>E-8</i>	<i>UNIT AND INTEGRATION VERIFICATION AND VALIDATION</i>	<i>304</i>
<i>APPENDIX F</i>	<i>REAL-TIME PROTOCOL (RTP) TESTING</i>	<i>307</i>
<i>F-1</i>	<i>OVERVIEW AND ASSUMPTIONS</i>	<i>309</i>
<i>F-1.1</i>	<i>Overview</i>	<i>309</i>
<i>F-1.2</i>	<i>Voice File Transfer and Analysis and Evaluation Assumptions.....</i>	<i>309</i>
<i>F-2</i>	<i>RTP TRANSFER TESTING</i>	<i>310</i>
<i>F-2.1</i>	<i>Detailed Implementation.....</i>	<i>310</i>
<i>F-2.2</i>	<i>Hardware Setup</i>	<i>310</i>
<i>F-2.3</i>	<i>Data Path.....</i>	<i>311</i>
<i>F-2.4</i>	<i>Software Setup</i>	<i>311</i>
<i>F-2.4.1</i>	<i>VQuad Setup</i>	<i>311</i>
<i>F-2.4.2</i>	<i>Modify or Create a VQuad Device Configuration for RTP Testing</i>	<i>314</i>
<i>F-2.4.3</i>	<i>Auto-Traffic Configuration</i>	<i>316</i>
<i>F-2.4.4</i>	<i>Configure Device 1</i>	<i>317</i>



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Table of Contents - Continued

<i>F-2.5 Modify or Create a VQuad Auto Traffic Configuration for RTP Testing...</i>	<i>321</i>
<i>F-2.6 VQT Setup</i>	<i>323</i>
<i>F-2.7 Checking VQT Auto-Measurement Setup</i>	<i>329</i>
<i>F-3 TEST EXECUTION</i>	<i>331</i>
<i>F-3.1 Preparation</i>	<i>331</i>
<i>F-3.2 Start VQT</i>	<i>332</i>
<i>F-3.3 Start VQuad</i>	<i>332</i>
<i>F-4 TRANSFER/TEST MONITORING</i>	<i>333</i>
<i>F-4.1 VQuad Status Monitoring</i>	<i>333</i>
<i>F-4.2 VQT Testing Monitoring</i>	<i>334</i>
<i>F-5 TEST RESULTS</i>	<i>338</i>
<i>F-5.1 PAMS</i>	<i>339</i>
<i>F-5.2 PSQM and PSQM+</i>	<i>340</i>
<i>F-5.3 PESQ</i>	<i>340</i>
<i>F-6 UNIT AND INTEGRATION VERIFICATION AND VALIDATION</i>	<i>340</i>
<i>APPENDIX G REFERENCES</i>	<i>343</i>
<i>SYMBOLS, ABBREVIATIONS, AND ACRONYMS</i>	<i>355</i>



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

List of Illustrations

TABLE 2-1 FEATURES SUPPORTED SPHERICALL IP PBX RELEASE 5	12
FIGURE 2-1 SIP CLIENT REGISTRATION	24
FIGURE 2-2 EXAMPLE OF REGISTRATION	25
FIGURE 2-3 BASIC SIP CALL	26
FIGURE 2-4 SIP MESSAGE DETAILS	28
FIGURE 2-5 SIMPLE CONFERENCE CALL WITH SIP	30
FIGURE 2-6 EXAMPLE CONFERENCE USING A PROXY	31
FIGURE 2-7 INVITE MESSAGE	36
FIGURE 2-8 CALL PARK	37
FIGURE 2-9 PARK SERVER 1	37
FIGURE 2-10 PARK SERVER 2	39
FIGURE 2-11 DIRECTED CALL PICK-UP	41
FIGURE 5-1 INSTAVOIP	103
FIGURE 6-1 UNPROVISIONED COUGAR WIN32 DEMO	113
FIGURE 6-2 INITIAL PROVISIONING DISPLAY	114
FIGURE 6-3 PROVISIONING NETWORK SETUP	115
FIGURE 6-4 STATIC IPV4 ADDRESSING	116
FIGURE 6-5 VOIP ACCOUNT SETTINGS	117
FIGURE 6-6 MINIMUM VOIP ACCOUNT SETTINGS	118
FIGURE 6-7 VOIP MEDIA SETTINGS	119
FIGURE 6-8 SECURITY SETTINGS	120
FIGURE 6-9 ADVANCED SETTINGS	121
FIGURE 6-10 VOIP XML PERSISTENT DATA	122
FIGURE 6-11 COUGAR WIN32 DEMO REGISTERED	123
FIGURE 6-12 WIRESHARK – SIP REGISTER	124
FIGURE 6-13 COUGAR WIN32 DEMO – MAKE CALL	125
FIGURE 6-14 COUGAR WIN32 DEMO – RING-BACK	126
FIGURE 6-15 WIRESHARK VOIP TELEPHONY ANALYSIS – SIP CALL	127
FIGURE 6-16 VISUAL STUDIO – HEADLESS CLIENT SOLUTION	129
FIGURE 6-17 HEADLESS CLIENT – SIP CONFIGURED	130
FIGURE 6-18 HEADLESS CLIENT – ACTIVE CALL	131
FIGURE 6-19 HEADLESS CLIENT – GUI ACTIVE CALL	132
FIGURE 7-1 LOOPBACK TRANSFER TEST SETUP	149
FIGURE 7-2 LOOPBACK SETUP	149
FIGURE 7-3 LOOPBACK_20x1_MASTER AUTO-TRIGGER TAB	159
FIGURE 7-4 LOOPBACK_20x1_MASTER GENERAL TAB	160
FIGURE 7-6 LOOPBACK_20x1_MASTER RX PROVISIONING TAB	161
FIGURE 7-7 LOOPBACK_20x1_SLAVE AUTO-TRIGGER TAB	163
FIGURE 7-8 LOOPBACK_20x1_SLAVE GENERAL TAB	164



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

List of Illustrations - Continued

FIGURE 7-9 LOOPBACK_20x1_SLAVE TX PROVISIONING TAB	164
FIGURE 7-10 LOOPBACK_20x1_SLAVE RX PROVISIONING TAB	165
FIGURE 7-11 PC-TO-PC RTP TRANSFER TEST SETUP	182
FIGURE 7-12 PC-TO-PC TRANSFER TEST SETUP	183
FIGURE 7-13 PCX_TRANSFER_20x1 AUTO TRIGGER TAB	190
FIGURE 7-14 PCX_TRANSFER_20x1 GENERAL TAB	190
FIGURE 7-15 PCX_TRANSFER_20x1 TX PROVISIONING TAB	191
FIGURE 7-16 PCX_TRANSFER_20x1 RX PROVISIONING TAB	191
FIGURE 7-17 SIP PHONE – SIP PHONE TRANSFER TEST SETUP	194
FIGURE 7-18 PCX_ENDToEND_20x1 AUTO-TRIGGER TAB	200
FIGURE 7-19 PCX_ENDToEND_20x1 GENERAL TAB	201
FIGURE 7-20 PCX_ENDToEND_20x1 TX PROVISIONING TAB	201
FIGURE 7-21 PCX_ENDToEND_20x1 TX PROVISIONING TAB	202



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Abstract

The objective of this research paper is to detail our exploration of open source SIP solutions as a viable alternative to building an Assured Services SIP end instrument (aka IP phone) as defined by the DOD Unified Capabilities and requirements specification.

This report is based on the following research and validation testing:

- L-3 Maritime Systems internal research,
- L-3 Maritime Systems VoIP lab research,
- L-3 Maritime Systems prototyping of a VoIP Communications Terminal,
- Unified Capabilities Requirements 2008 Change 1,
- L-3 Maritime Systems Technical Report *Intelligent Advanced Communications IP Telephony Feasibility for the US Navy*, Volumes 1 and 2 (ISRN L3COM/Maritime Systems/TR-2007/001),
- L-3 Maritime Systems Technical Report *Intelligent Advanced Communications IP Telephony Feasibility for the US Navy*, Phase 2 (ISRN L3COM/Maritime Systems/TR-2009/001),
- OSIP, an open source SIP stack,
- SipXtapi, an open source SIP stack,
- SipXtapi-3.3.0, an open source SIP stack,
- REDCOM Slice2100 soft switch with Transip, HDX,
- GL Communications VoIP quality test tool,
- Wireshark,
- Licensing requirements and legal implications of various open source VoIP solutions.

The major findings are:

- SIP stack behavior and characteristics varies substantially among vendors.
- Unlike traditional TDM based phones, SIP stacks and consequently SIP phones are somewhat switch specific in terms of compatibility and vary substantially in terms of what they chose to support. Though one can plug a TDM-based phone into another TDM network and base functionality will work, the same is not true for IP phones.
- 2 of the 3 open source SIP stacks we evaluated were maintained outside the United States.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- The SipxTapi package in particular has over 13,000 source files spread out over 2,500 folders. Vetting the source for potential malware and or back doors or any type of security risk would be extremely difficult.
- None of the open source stacks we surveyed had support for IPV6 fully integrated with the SIP stack or at all.
- IP based voice and video networks are already in wide use today in networks where there is little or no control over competing network traffic. A closed network environment on board a ship would provide an optimal environment for IP based communications.
- A stack based on OSIP would likely be the best candidate for developing a UCR defined SIP end instrument for the Navy or any DOD entity, especially for host based solutions



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

CHAPTER 1 Executive Summary

1.1 Objectives

The objective of this research paper and the Proof-of-Concept IP Communications Terminal Software is to research technologies and solutions supporting the communications stack necessary to implement an assured services session initiated protocol, unified VoIP (IP telephony), Video, and Data infrastructure on US Navy vessels.

This report is based on the following research and validation testing:

- L-3 Maritime Systems internal research,
- L-3 Maritime Systems VoIP lab research,
- L-3 Maritime Systems prototyping of a VoIP Communications software,
- Unified Capabilities Requirements 2008 (UCR),
- Unified Capabilities Requirements 2008 Change 1
- Government documents focusing on network security and implementation,
- RFC,
- The L-3 Henschel Technical Report *Intelligent Advanced Communications IP Telephony Feasibility for the US Navy*, Volumes 1 and 2 (ISRN L3COM/HENSCHTEL/TR-2007/001) and
- The L-3 Henschel Technical Report *Intelligent Advanced Communications IP Telephony Feasibility for the US Navy*, Phase 2 Volumes 1 (ISRN L3COM/HENSCHTEL/TR-2009/001).

Unless noted, all references to the UCR pertain to the UCR Change 1.

1.2 Introduction

This document is the third in a series of research papers that laid the foundation for the implementation of the Assured Services (AS) Session Initiated Protocol (SIP) (AS-SIP) stack, as well as the foundation for the methods used to test the network for Mean Opinion Scores. Our research is separated into two documents: Volume I discusses General Implementation, the use of Open Source, and network testing. Volume II is dedicated to the Embedded Proof-of-Concept Development process.

The previous feasibility study and subsequent investigation resulted in the following “lessons learned”:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

a. Variations between RFC and UCR – The requirements between the UCR and the RFC are different to the point that the standard SIP stack requires an extensive amount of recoding to accomplish the requirements of the UCR.

b. The network is the critical piece – If the network is not configured and sized for current and future requirements, the Mean Opinion Score (MOS) will not be acceptable.

c. The US Navy needs reliability over technology – Tactical functions must be assiduous and available at all times.

d. VoIP has become mission critical – VoIP is being used commercially in telephone structures, and on military bases around the world. It has already been approved by several military organizations including JITC.

e. Planning and testing up front is critical – Understanding owners as well as ownership of data on the network; and understanding the need to work together to develop the plan for QoS and reliability for the complete network is of utmost importance.

Our previous investigations also revealed the following findings for implementing a unified VoIP (IP telephony), Video and Data infrastructure on US Navy vessels:

- Assured Services Session Initiation Protocol (AS-SIP) can be implemented on an end device,
- The UCR varies from the RFC making a compliant SIP stack non-functional for an AS-SIP implementation,
- The implementation from the UCR is open to interpretation, meaning that end devices need to have variations in their implementation of the stack,
- Switch manufacturers have implemented AS-SIP for trunks, but have limited functionality for end devices
- The Proof-of-Concept IP Communications Terminal can handle calls as well as other network tasks, bringing communications and data together into a unified device
- Implementing an IP solution on US Navy vessels is feasible and achievable
- The US Government is assessing the benefits of Open-Source versus proprietary vendor offerings
- The unification of VoIP (IP telephony), video, and data will be secure as detailed in US Department of Defense publications
- SIP stack behavior and characteristics vary substantially among vendors resulting in differences in traditional TDM based phones, and SIP stacks. Consequently SIP phones are switch-specific in terms of compatibility and vary substantially in terms of what they choose to support. Though one can plug a TDM-based phone



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

into another TDM network and base functionality will work, the same is not true for IP phones.

- The SipxTapi package in particular has over 13,000 source files spread out over 2,500 folders. Vetting the source files for potential malware and or back doors or any type of security risk, would be extremely difficult. Two of the three open source SIP stacks we evaluated were maintained outside the United States.
- None of the open source stacks we surveyed had support for IPV6 fully integrated with the SIP stack (or at all).
- IP-based voice and video networks are already in wide use today in networks where there is little or no control over competing network traffic. A closed network environment on board a ship would provide an optimal environment for IP based communications.
- A stack based on OSIP would likely be the best candidate for developing a UCR defined SIP end instrument for the Navy or any DOD entity, especially for host based solutions.
- There will continue to be major investments in this infrastructure under an Open Systems Architecture consortium, but with the differences between AS-SIP and SIP implementation; COTS products will be limited until the manufactures are forced to develop them.
- The government is assessing the benefits of Open Source versus proprietary vendor offerings. Open Source solutions reduce the upfront costs, but may have longer term support costs.
- The unification of VoIP (IP telephony), video and data can be secure as detailed in US Department of Defense publications. Security methods must be investigated for VoIP differently than other network data for performance as well as the resulting relationship to the mean opinion score (MOS).

Much of the information and the findings for our embedded prototype are identical to that of the open source stacks we developed. The fact that the *Unicoi* SIP stack was embedded was irrelevant in terms of the performance of the stack although the embedded approach is inherently more secure due to the nature of the embedded environment.

At the beginning of 2011, the updated UCR 2008 released Change 2. Since the source development for the open source and embedded prototype were completed in the documentation mode, the changes in the Change 2 version are not reflected within this document.



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

This page intentionally left blank.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

CHAPTER 2 VoIP Review

2.1 Overview

The Internet landscape of today borrows immensely from the research funded by the Advanced Research Project Agency (ARPA), later renamed Defense Advanced Research Projects Agency (DARPA). Among DARPA's initial objectives was to foster communications among geographically distributed computers. One of the results was the Network Voice Protocol (NVP) that enabled transporting "real-time full-duplex digital voice over packet switched computer networks" [1] as in the Advanced Research Projects Agency Network (ARPANET), which became the precursor of today's Internet. NVP was first implemented in December 1973 and was used to send speech between ARPANET sites. [1]

VoIP entrepreneurs noted the business potential of sending voice data packets over the Internet rather than communicating through standard telephone service to avoid long distance charges. Vocaltec introduced its Internet phone software in 1995. This software used the H.323 protocol and ran in personal computers (PCs) with sound cards, microphones, and speakers. Both the caller and the called PC needed to have a computer equipped with the same software and the same kind of soundcard with the latest drivers installed. By 1998, PC to phone service was in place and phone-to-phone service soon followed with a computer establishing the connection. VoIP traffic grew to approximately 1% of all voice traffic in the United States by 1998. By 2000, VoIP traffic accounted for more than 3% of all voice traffic. Once hardware started becoming more affordable, larger companies were able to implement VoIP on their internal IP networks, and long distance providers even began routing some of the calls on their networks over the Internet.

2.2 VoIP Usage

Since 2000, VoIP usage has expanded dramatically. The standards and research consortia, such as the Internet Engineering task Force (IETF), International Telecommunications Union (ITU), and CableLabs, worked on defining standards and interoperable VoIP solutions. Over the years, many different technical standards for VoIP packet transfer and switching has been created such as, H.323 [1], Simple Gateway Control Protocol (SGCP) [1], IP Device Control (IPDC) [1], Media Gateway Control Protocol (MGCP) [1], Megaco/H.248 [1], Data Over Cable Service Interface Specification (DOCSIS), and SIP (Session Initiation Protocol) [1], just to name a few of the major ones. At present it seems that vendors are converging towards accepting SIP as the premier next generation VoIP protocol for its simplicity and adaptability. However that comes with the price: there are quite a few SIP extensions and SIP is notorious for having several ways to implement any given service. In just a few short years, VoIP has gone from being a fringe R&D topic, to a viable alternative to standard telephone service.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Given the worldwide move towards IP technology, the rapid acceptance of VoIP, and the pace of development of multiple applications that are quickly being integrated over IP, it is recommended without any reservation that a multi-theater and multi-location enterprise such as the US Navy, should place emphasis on introducing multi-services networks over IP, and assess its benefits as well as its limitations in order to expand its services as the tests prove to be satisfactory. Below is an abbreviated timeline to show how fast VoIP has moved from a laboratory experiment to a viable telephony service:

Timeline: [1 unless otherwise noted]

1973 – NVP

1995 – Vocaltec - First Internet phone

1996 – H323 is approved [3]

1996 – SIP is designed designed by Henning Schulzrinne (Columbia University) and Mark Handley (UCL) [4]

1996 – Internet phones catch the attention of US telecommunication companies who ask the US Congress to ban the technology, Telecommunications act of 1996 the beginning of convergence.[2]

1998 – 1% of all voice traffic

1999 – SIP RFC 2543 ratified [6]

2000 – 3% of all voice traffic

2002 – SIP RFC 3261 is ratified [5]

2005 – 16 million users and 5 billion dollar revenue

2009 – est. 55 million users and 17 billion dollar revenue

2.3 VoIP Protocol Suites

At present, two major application layer protocol suites dominate multi-service communications over IP: SIP and H.323. They offer very different approaches to application signaling and services. H.323 started earlier in the mid 1990's but SIP is rapidly becoming the application protocol of choice, specifically for VoIP, for its scalability and adaptability. However, along with the ease comes the issue of uniformity in implementation. SIP is notorious for having several ways of implementing any given service. Mostly for this reason, many manufacturers support SIP natively with a restricted set of features and also use it purely as a signaling protocol over which their proprietary protocols are run. Alcatel-Lucent for example, supports both SIP and H.323. H.323 is used as a transport for its proprietary protocol "Universal Alcatel (UA)". SIP is supported natively with a set of basic telephony features and also as a signaling protocol in combination with Hyper-Text Transfer Protocol (HTTP) and Voice eXtensible Markup Language (VXML) with softphones. As SIP's capability in supporting a larger set of telephony features natively is enhanced with extensions through the efforts and support of the Internet Engineering Task Force (IETF) and other Internet communities, the features will be incorporated with native SIP.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

H.323 is based heavily on the ITU multimedia protocols that preceded it, including H.320 for Integrated Services Digital Network (ISDN), H.321 for Broadband Integrated Services Digital Network (BISDN), and H.324 for General Switched Telephone Network (GSTN) terminals. The encoding mechanisms, protocol fields, and basic operation are somewhat simplified versions of the Q.931 ISDN signaling protocol.

The Session Initiation Protocol (SIP) [1], developed in the Multiparty Multimedia Session Control (MMUSIC) working group of the IETF, takes a different approach to Internet telephony signaling by reusing many of the header fields, encoding rules, error codes, and authentication mechanisms of HTTP.

SIP is becoming the protocol of choice for the application layer for its extensibility, scalability, and adaptability. For the same reasons, different SIP extensions indicate different ways of implementing any given service, which can be extremely vexing in the user and the vendor community. Relevant SIP protocol suites with extensions should be tested at Department of Defense (DoD) test laboratories and the Defense community should be a major driver for pushing a common, interoperable set of SIP features.

2.4 Session Initiation Protocol

The Session Initiation Protocol (SIP) is an application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants without dependency on the type of session that is being established. These sessions include Internet telephone calls, multimedia distribution, and multimedia conferences. SIP itself does not participate in the sessions themselves but merely enables and controls them. There are several other protocols that have been developed to do the actual work of carrying the session data.

As it stands today, not many extensions have been standardized. Several proposals have been introduced that would provide additional functionality for telephony applications but as of yet they have not been ratified. Because only a few extensions have been ratified, SIP on its own can only perform a few basic functions as it relates to telephony. It is for this reason that vendors have had to develop their solutions to this problem while they wait for the protocol to mature. However, this does not prevent the use of SIP today.

SIP is a request-response protocol that closely resembles two other Internet protocols, HTTP and SMTP (the protocols used for the worldwide web and email); consequently, SIP sits comfortably alongside Internet applications. Using SIP, telephony becomes another web application and integrates easily into other Internet services. SIP is a simple toolkit that service providers can use to build converged voice and multimedia services. However, it is crucial to understand that other protocols must be used alongside SIP in order to provide complete telephony services.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.4.1 Components in a SIP Environment

In the simplest SIP environment, the only two required components in SIP transaction are a User Agent Client and a User Agent Server. To achieve a complete communications system there are several other dedicated server applications. These server applications may be individual applications on individual computers, or be integrated into a few applications housed on a limited number of computers. The number of applications and computers is dependent on the vendor, size of the implementation, and the requirement for redundancy in the final solution. The following paragraphs are a list of the more popular server applications, some of which are defined in the specifications, and others are implemented to serve a predetermined need, which is not defined directly in the different specifications.

2.4.1.1 User Agent Client (UAC)

The UAC is the caller, typically a hard or soft phone. “A User Agent Client is a logical entity that creates a new request, and then uses the client transaction state machinery to send it. The role of UAC lasts only for the duration of that transaction. In other words, if a piece of software initiates a request, it acts as a UAC for the duration of that transaction. If it receives a request later, it assumes the role of a User Agent Server for the processing of that transaction.¹”

2.4.1.2 User Agent Server (UAS)

The UAS is the callee, which could be another phone or server. “The user agent server is a logical entity that generates a response to a SIP request. The response accepts, rejects, or redirects the request. This role lasts only for the duration of that transaction. In other words, if a piece of software responds to a request, it acts as a UAS for the duration of that transaction. If it generates a request later, it assumes the role of a User Agent Client for the processing of that transaction.²”

2.4.1.3 Back-to-Back User Agent (B2BUA)

“A Back-to-Back User Agent (B2BUA) is a logical entity that receives a request and processes it as a User Agent Server (UAS). In order to determine how the request should be answered, it acts as a User Agent Client (UAC) and generates requests. Unlike a proxy server, it maintains dialog state and must participate in all requests sent on the dialogs it has established. Since it is a concatenation of a UAC and UAS, no explicit definitions are needed for its behavior.³”

¹ RFC 3261, <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007

² RFC 3261, <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007

³ RFC 3261, <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.4.1.4 Location Server

From RFC 3261, “A location service is used by a SIP redirect or proxy server to obtain information about a callee's possible location(s). It contains a list of bindings of address-of-record keys to zero or more contact addresses. The bindings can be created and removed in many ways; this specification defines a REGISTER method that updates the bindings.⁴”

In simple terms, this is a lookup service allowing one to resolve an identity to an (optimized) list of URIs for the contact.

2.4.1.5 Outbound Proxy

An Outbound Proxy is defined as “A proxy that receives requests from a client, even though it may not be the server resolved by the Request-URI. Typically, a UA is manually configured with an outbound proxy, or can learn about one through auto-configuration protocols.⁵” An outbound proxy is typically used when one wishes all the outgoing calls to go through a proxy, typically because external rules need to be applied to the outbound call. This would be common in the case where you were using a media gateway to convert from SIP to PSTN.

2.4.1.6 Media Gateway (MG)

“A Media Gateway acts as a translation unit between disparate telecommunications networks such as PSTN; Next Generation Networks; 2G, 2.5G and 3G radio access networks or PBX. Media Gateways enable multimedia communications across Next Generation Networks over multiple transport protocols such as ATM and IP. Because the MG connects different types of networks, one of its main functions is to convert between the different transmission and coding techniques. Media streaming functions such as echo cancellation, DTMF, and tone sender are also located in the MG.⁶”

2.4.1.7 Presence Server

A presence server allows users to update their presence information that is then shared with others on the network. This presence information can allow others to see your availability, and contact you appropriately. SIP Instant Messaging (IM) Presence is an extension that provides Instant Messaging functionality alongside availability information.

⁴ RFC 3261, <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007

⁵ RFC 3261, <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007

⁶ Media Gateway – Wikipedia, the free encyclopedia,
http://en.wikipedia.org/wiki/Media_Gateway, July 6, 2007



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.4.1.8 Proxy Server

A Proxy Server is defined as “An intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients. A proxy server primarily plays the role of routing, which means its job is to ensure that a request is sent to another entity "closer" to the targeted user. Proxies are also useful for enforcing policy (for example, making sure a user is allowed to make a call). A proxy interprets, and, if necessary, rewrites specific parts of a request message before forwarding it⁷”

2.4.1.9 Redirect Server

“A redirect server is a User Agent Server that generates 3xx responses to requests it receives, directing the client to contact an alternate set of URIs.⁸” It accepts SIP requests, maps the address into new addresses which are then returned to the client and provides information on callers’ redirect and proxy servers.

2.4.1.10 Registration Server

From RFC 3261, “A registrar is a server that accepts REGISTER requests and places the information it receives in those requests into the location service for the domain it handles.⁹”

In simple terms, clients register with the registration server when they come online, giving it information about their identity, how to reach them, and who they will allow to communicate with them. This info is then stored in the same database that the Location Server uses for client lookups.

2.4.1.11 Session Border Controller (SBC)

“A Session Border Controller is a device used in some VoIP networks to exert control over the signaling and usually also the media streams involved in setting up, conducting, and tearing down calls.¹⁰”

A session border controller is normally used at the border of a company’s network along with a firewall to enforce company security and network policies. A Session Border Controller can act as either a Back-to-Back User Agent or a proxy, both restricting and

⁷ RFC 3261, <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007

⁸ RFC 3261, <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007

⁹ RFC 3261, <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007

¹⁰ Session Border Controller- Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Session_Border_Controller, July 6, 2007



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

changing the contents of the SIP packets. Some interesting uses of SBCs include the ability to record calls leaving an organization and allowing transcoding due to incompatible codecs.

2.4.1.12 Simple Traversal of UDP Through NATs Server

A Simple Traversal of UDP Through NATs (STUN) server is used typically by a UAC to determine its public IP address and help identify the type of firewall Network Address Translators (NAT) in place. This information is used to help setup UDP communications for a call behind a NAT'd endpoint.

2.5 SIP Limitations

SIP was designed to solve a small but important set of issues and to allow interoperability with a broad spectrum of existing and future IP telephony protocols. To this end SIP provides four basic functions:

1. User Location: mapping a user's name to their current network address (similar to DNS)
2. Feature Negotiation: Allows User Agents (UA) to negotiate a common set of features
3. Call participant management: adding, dropping, or transferring participants
4. Modifying session features while a call is in progress.

Any other functions must be performed by other protocols.

Its simplicity means that SIP is not a Session Description Protocol (e.g., SDP) nor is it able to perform Conference Control functions. It is also not a Resource Reservation Protocol (e.g., RSVP) and it has nothing to do with guaranteeing quality of service (QoS) (e.g., 802.1p, Type of Service (TOS)). SIP can work within a framework with other protocols to insure these roles are played out - but SIP does not perform these functions itself. SIP is regularly deployed alongside SOAP, HTTP, XML, VXML, WSDL, UDDI, SDP, RTP and a variety of other protocols.

Because of the simple nature of SIP many of the functions under study are not achievable with native SIP alone. Standard methods for deploying these features have not yet been ratified due to the myriad of different potential methods to deploy any given feature. Vendors have independently chosen varying methods to solve these issues, which, in turn create a non-interoperable or proprietary situation. Due to this current situation where vendors have not reached agreement (as well as the the lack of standards) it will be many years before multi-vendor solutions with “plug-and-play” advanced features are available to the marketplace. This leads to an environment today where COTS SIP products cannot be guaranteed to interoperate or even have similar feature sets.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.5.1 Current Supported features

Table 2-1 is an abbreviated list of features that are supported by NEC Sphere's Sphericall IP PBX in release 5.¹¹ JITC-certified Sphericall IP PBX delivers assured connectivity via MLPP for special C2 (Command and Control) users including strategic leadership and those who manage strategic assets. The list is very extensive, compared to the few items that were determined to need to be implemented above and supported in the SIP Feasibility section and appendix.

Table 2-1 Features Supported Sphericall IP PBX Release 5

General Telephony Services
Call Announce
Call Transfer
Call Coverage: Multi-Level, Follow Me, Conditional
Call Forward
Call Hold
Call Waiting
Music-On-Hold
On-Hold Reminder
Dial-Out Authorization Codes
Direct Inward Dial
Direct Outward Dial
Inbound Routing Schedules (Automatic)
Message Waiting Indicators
Multi-Party Conferencing
Park Zones
Pickup Groups
Class Of Service Profiles
Permission Lists: Allow / Disallow Specific Numbers
Trunk Hunt Groups: Directional
User Access Authorization Codes
Automatic Route Selection (ARS)
Call Recording (Optional)
Call Admission Control
Multi-Level Precedence and Preemption for Emergency / Critical Communications
Call Accounting
Call Detail Reporting
Data Export: Originator ID, Receiver ID, Intended Receiver ID, Time, Duration, Outcome, Reason

¹¹ http://www.necsphere.com/solutions/Sphericall_Data_Sheet_53106.pdf



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Key Industry Standards Support
SIP - RFC 2543 / 3261
MGCP - RFC 2705 / 3149
SIPConnect for SIP Trunking
SIMPLE (Windows Messenger)
TAPI 3.0
DirectX 8.0
SMDI
TCP / IP / UDP
DHCP
FTP / TFTP
SNTP
RTP / RTCP
SOAP
XML
Advanced Communications Features
Call Recording (Optional)
Multi-Level Precedence and Preemption for Emergency / Critical Communications
Softphone for Mobile and Remote Users
User Presence Status Monitoring
Standard Telephony Features*
Caller ID Display
Call Transfer (Attended or Unattended)
Mute
Hold
Park / Unpark
Do Not Disturb
Transfer to Voice Mail
Redial
Incoming Call Indication with Caller ID
Message Waiting Indication
Missed Call Indication with Caller ID
Multi-Party Audio Conferencing
* Phone/device dependent.

A key group of features were chosen to be reviewed in detail as to how they would be implemented in SIP, and if the protocol supported the feature directly. The features chosen were ones that were determined as important to an installation on a US Navy



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

vessel. There were other features which were not evaluated because they were known to already be implemented or were not required.

2.5.2 Selected SIP Features

A group of features that are not documented were chosen for a more detailed review. These features were chosen as possible requirements in different US Navy internal communications systems. Each feature is listed independently and with a description. A section is included that explains the changes that may be required in SIP-implementing companies, such as Sphere and Pingtel, who already reside under the umbrella of the SIP specifications and drafts and have already implemented many of these features.

2.5.2.1 Conference Call

A conference call is the creation of a group of end devices coupled together so all participants can hold a conversation. It requires action of the end user to pick up their end device or call into a conference number.

SIP was defined to allow for the establishment, maintenance, and termination of calls between one or more users. However, despite its origins as a large-scale multiparty conferencing protocol, SIP is used today primarily for point-to-point calls. This configuration is the focus of the SIP specification and most of its extensions. As a result, there is a lot of confusion about how SIP supports multi-party conferencing.

There are a number of conferencing models supported by native SIP. These models range from Three-Party Calling with end system mixing to large multicast conferences, dial-in or dial-out conferences servers, to ad-hoc centralized conferences, and to conferences using centralized signaling and distributed media. Most conference calls involving more than a dozen or so participants have the need for more advanced features such as the ability of the moderator to mute all phones, set the length of time for the conference call, record and display the participants of the conference call, etc. However, there is no ratified standard for any of these models as of yet. A draft exists by Rosenberg, titled *draft-rosenberg-sip-conferencing-models-00* that describes all of these models in detail. There are products available that work with different vendor solutions. All the vendors confirmed they had a method to support it.

Changes Needed to SIP Protocol

No changes or modifications of SIP are required to implement the features, however, a User Agent Server (UAS) or mixer is required.

2.5.2.2 Intercom

This feature involves the creation of a group of end devices coupled together so all participants can hold a conversation. It doesn't require the end user to pick up their end device or hang it up. The end device goes off-hook automatically and then returns to on-hook after the intercom is completed.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The Session Initiation Protocol does not currently provide a mechanism to force the UA (User Agent) to go *off-hook*. A UA could be configured to *auto-answer* incoming calls. It is possible, and some manufacturers have implemented this method, to add a field to the standard INVITE header that would cause the receiving UA to go off-hook automatically with or without mute. In essence, the called device must understand this field or flag and it must be programmed to act on it. A number of manufacturers have successfully implemented this feature. One of the end devices that support it is Polycom; they use a parameter in the INVITE SIP header to signal the phone. This method can be used but must be implemented in the IP-PBX as well as the end device. To use the phone for announcing, the Pager Server must also support the method. There is an Internet draft called “draft-ietf-sip-answermode-00” but it expired in June of 2006 with no action taken. The creation of a group of end devices so all participants can hold a conversation is in essence a form of Conference Call.

Changes Needed to SIP Protocol

Methods to extend SIP have been implemented by some manufacturers but an Internet draft must be proposed, discussed and ratified in order to provide this function in a standard manner and guarantee availability in any given SIP device.

2.5.2.3 Group Page

The creation of a group of end devices coupled together so a one-way announcement can be done. The end device goes off-hook automatically and then returns to on-hook after the announcement is completed.

This feature has similar issues to the Intercom feature described above in that it involves calling a group of users simultaneously. Like the Intercom feature above, it would also be an invite-based conference call. The basic difference between these features is that the Intercom feature calls for a two-way conversation and Group Page calls for a one-way conversation but the mechanics are identical. The native SIP vendors that didn't support it indicated that it could be added to their product line. This can also be done by the Announcing system, or a conjunction between the two systems.

Changes Needed to SIP Protocol

An auto-mute function must be added to SIP.

2.5.2.4 Priority Calls

The ability for an individual to break into a call that is in progress by use of a feature code and supervisor login. SIP, as a signaling protocol, does not have the ability to break into ongoing calls. The support for Multi-Level Precedence and Preemption (MLPP), can be used instead, see below.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Changes Needed to SIP Protocol

Extensions to SIP would need to be drafted and ratified in order to provide this function. The SIP protocol would need to have the ability to signal the UA and have the UA duplicate the incoming and outgoing voice streams to add the new UA into a new conference call.

2.5.2.5 NCS Voice Precedence System

Known as Multi-Level Precedence and Preemption, this feature gives individuals the ability to override a call that is in progress between two or more parties. This feature is presently not being used onboard US Navy vessels. Instead, Priority Calling is being used. Priority Calling differs from MLPP in that it is a method of inserting a third-party into an ongoing call without notification to the original parties. An operator or similar person traditionally used it to check the status of the line. MLPP on the other hand, is a priority-based call override system. This system may affect all SIP elements in a network. It is identical to a basic call with the addition of the *Resource-Priority* field in the INVITE message. Not all vendors have implemented this feature but it has been implemented by Sphere for their JITC certification.

Changes Needed to SIP Protocol

No changes needed to SIP. At a minimum, support of RFC 4412 is required by the UA's. Preferably, all SIP elements should support this RFC.

2.5.2.6 Call Park

The ability to park a call onto a virtual extension and then have a third-party or the same individual, retrieve the call. To the caller, the call appears to be on hold and is presented with “music on hold”, dependent on the implementation.

This feature, like many others, can be implemented with SIP in several ways. It is defined in RFC 4240 and in the *draft-procter-sipping-call-park-extension-00* draft document. Several different methods are also described in books and SIP-related sites on the Internet. It has been implemented by both of the native SIP vendors.

Changes Needed to SIP Protocol

No changes are required for this method. The only requirement is a “Park Server”. Other methods may require modifications and/or additional servers.

2.5.2.7 Directed Call Pick-up

This feature provides the ability to pick-up a call that is ringing on another end device. Call Pickup is described in the IETF draft called *draft-worley-sipping-pickup-02*. This document states:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

“There are several different schemes for implementing call pickup. The basic method is the one specified in the Sylantro "SIP-B" specification, which despite its proprietary air, uses standard SIP features in an end-point call control (EPCC) style. All other methods are variations on the same theme, usually by using an agent process (in a proxy or communications server) to provide a feature that the user agents are lacking. Like call transfer, effecting call pickup requires some support from the caller's end. These caller-end features will, therefore, soon come to be considered necessary for any "quality" SIP implementation.” It has been implemented by both of the native SIP vendors.

Changes Needed to SIP Protocol

No changes to the protocol itself are required. Some of the more complex implementations, in particular those requiring proxies and servers, will require software additions. It would be beneficial to have a RFC so a single method is used across multiple vendors' in their implementations

2.5.2.8 Group Call Pick-up

The SIP flows for this function are identical to the flows required for Directed Call-Pickup (above). The exception being that the called extension is a conference URI. In other words, the group of end-devices must be in a conference. You may then poll the conference URI to obtain the ringing party's URI. It has been implemented by both of the native SIP vendors.

Changes Needed to SIP Protocol

No changes to the protocol itself are required. Some of the more complex implementations, in particular those requiring proxies and servers, will require software additions.

2.5.2.9 Recording of Calls

This feature would provide the ability to record calls from a set of pre-defined end devices as soon as they are off hook. This would be used for all calls that the bridge handles.

There are a number of ways in which this feature might be implemented. The community has discussed this subject extensively since the year 2000. Several ideas have been discussed but no drafts or standards have come of it. The potential implementations would all require the creation of a Conference Call and the automatic addition (or INVITE) of a *recorder*. The *recorder* would simply act as a standard UA and store the mixed streams it receives. Another method that some companies use is a promiscuous-mode node that looks at the SIP traffic on the network and records calls per a pre-defined filter that captures the packets for a particular call. This method has no requirements of the SIP protocol; other than its headers must conform to the RFC 3261 specification. The securing of the control signal and the bearer streams complicates this even further. The



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

recording device must be able to decrypt the stream or it must be decrypted by a Session Board Controller.

Changes Needed to SIP Protocol

No changes to SIP need be implemented. However, a standard should be proposed and ratified that defines the methodology for achieving this functionality. Until a standard is available there is no guarantee any two vendors will implement this feature in the same way.

2.5.3 Selected SIP Features Summary

After extensive research, the salient point is simple: all of the above features have been or could be implemented with SIP with approximately 90% of the above features implemented to the SIP standard or to draft versions of the standard. For any given feature there are a number of ways to achieve it and this is typical for new protocol implementations as they mature. This, coupled with SIP's simplicity, we believe, is the reason for the current state of any confusion with regards to SIP standardization. Several of the features such as Intercom have been implemented with Polycom phones; this method is well documented and can be added into the end devices that would be used. Group Paging also requires the auto-answer like the Intercom. The use of the Announcing system can also be used instead of group paging on the phone system.

Priority Calls can be achieved with the use of MLPP. None of these appear to be preventive issues for utilization, and in the future there will be ratified RFCs that will have support from multiple vendors. In addition, we can envision that eventually the SIP protocol standard will be completed in such a manner as to provide standard methods of implementation for these and other telephony features as the market matures, which is being lead by the readily available cost effective products being deployed in the business and home environments.

2.6 Session Initiation Protocol (SIP) Feasibility

2.6.1 Introduction to SIP

This section will examine the use of Session Initiation Protocol (SIP) for a number of telephony features within the context of the DoD needs and requirements. The primary goal is to define the feasibility of implementing these features using native SIP as the signaling protocol. This paper assumes that the term "native SIP" refers to the SIP protocol standards as defined by the Internet Engineering Task Force (IETF). In this way, we hope to determine whether these features can be implemented when using standard COTS SIP-based devices in a multi-vendor environment without modification. For any given feature that cannot be implemented as described above, a workaround solution is investigated. We strive to suggest workaround solutions that are themselves based on accepted IETF or IEEE protocol standards.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.6.2 What is SIP?

The Session Initiation Protocol (SIP) is an application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants without dependency on the type of session that is being established. These sessions include Internet telephone calls, multimedia distribution, and multimedia conferences. SIP itself does not participate in the sessions themselves but merely enables and controls them. There are several other protocols that have been developed to do the actual work of carrying the session data.

In recent years the community has chosen SIP as the de-facto signaling protocol for Voice over IP (VoIP) applications. It was designed specifically to be simple and highly extensible. It is this simplicity that makes it the perfect signaling protocol, as it is highly scalable and independent of media and session type. However, the protocol is still evolving today as the technology matures. As it stands today, not many extensions have been standardized. Several proposals have been introduced that would provide additional functionality for telephony applications but as of yet they have not been ratified.

Because only a few extensions have been ratified, SIP on its own can only perform a few basic functions as it relates to telephony. It is for this reason that vendors have had to develop their solutions to this problem while they wait for the protocol to mature. However, this does not prevent the use of SIP today. Most solutions available today use SIP for signaling but rely on other protocols, both proprietary and standard, to complement SIP and thereby achieve the more complex features.

SIP is a request-response protocol that closely resembles two other Internet protocols, HTTP and SMTP (the protocols used for the world-wide web and email); consequently, SIP sits comfortably alongside Internet applications. Using SIP, telephony becomes another web application and integrates easily into other Internet services. SIP is a simple toolkit that service providers can use to build converged voice and multimedia services. It is crucial to understand that other protocols must be used alongside SIP in order to provide complete telephony services.

This report provides only a rudimentary description of the protocol's workings. This description is provided only to place the work into context. A complete description of the components, specific extensions, and modes of operation of SIP is beyond the scope of this paper. For more information please see the protocol standard as defined by the Internet Engineering Task Force (IETF) in RFC3261. This RFC obsoletes RFC2543, which was the original SIP specification. RFC3261 defines SIP itself and 6 extensions or methods. The RFC is extended or amended by RFC3265, RFC3853 and RFC4320. These RFCs define additional extensions including SUBSCRIBE, NOTIFY, MESSAGE, INFO, SERVICE, NEGOTIATE and REFER. There are a number of additional RFCs related to SIP but the basic protocol is described by these four.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.6.3 Core SIP Specifications

These are the specifications that impact almost every session requested by an agent for which the extension is relevant, such as, SIP session management, SIP registrations and SIP subscriptions.

The specifications in this area are:

2.6.3.1 Fundamental SIP Protocol Related

- RFC 3261 – is the basis of SIP protocol.
- RFC 4320 – describes modifications to SIP RFC 3261 to address issues with SIP non-INVITE transactions.
- RFC 3263 – describes DNS procedures for taking a SIP URI and determining the IP address, port, and transport protocol for the SIP server that is associated with that SIP URI and may be in a different IP domain.
- draft-ietf-sip-connected-identity – The identity of the party answering a call, i.e., the connected user can differ from that of the initial called party (given in the ‘To’ header) in many services requiring forwarding and retargeting. This ID extends the use of the ‘From’ header field to allow it to convey the identity of the connected user. This is used in conjunction with authentication of the UA identity.

2.6.3.2 SDP Related

- RFC 4566 - Session Description Protocol (SDP), describes the format of multimedia sessions (e.g., VoIP conferences, voice and video streaming) for the purposes of session announcement, session invitation, and other forms of multimedia session initiation. It is independent of transport layer protocols. SDP conveys media details, transport addresses, and other session description metadata to the participants.
- RFC 3264 – defines an Offer/Answer model with the SDP, that is primarily used in unicast sessions with SIP to negotiate session parameters
- RFC 3605 – defines an extension of SDP to explicitly signal the IP address and port # for RTCP within an SDP message, rather than deriving from base media port in RTP. It is specially needed for NAT/NAPT, where mapped port numbers may have no relationship to the order of original port numbers.
- RFC 3388 – defines two SDP attributes, “group” and “mid” that allow grouping of media streams in SDP messages. This enables binding between different media streams from a single flow encoded in different formats (e.g., audio associated with a video feed) on different ports and host interfaces.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.6.3.4 SIP Requests Related

- RFC 3840 – Indicating User Agent (UA) Capabilities in SIP for example, in INVITE or REGISTER or OPTIONS requests, extend the capabilities in RFC 3261 by providing a general framework to indicate the characteristics and capabilities of SIP UA.
- draft-ietf-sip-outbound-07 – proposes changes to SIP registration mechanism to allow requests to be delivered across NAT bindings between Servers and User Agents (UA)
- draft-ietf-sip-gruu-11 – This SIP extension defines a mechanism (e.g., by SIP REGISTER) for obtaining and communicating an unique globally routable URI that can direct requests to a specific UA. This is usable for features like transfer.
- RFC 3265 – defines a general event notification framework in SIP with SUBSCRIBE and NOTIFY requests.

2.6.3.5 SIP Headers Related

- RFC 3325 – introduces a new header field, “P-asserted-identity”, that enable a network of trusted SIP servers in a trust domain to assert the identity of end users/systems and convey privacy indications, as in secure caller-id services
- RFC 4474 – introduces two new SIP header fields: 1) “Identity” to convey signatures in a cryptographic hash and “Identity-Info” for conveying a reference to the certificate of the signer for securely identifying originator UAs. It is an alternative to the mechanisms in RFC 3325 [59].
- RFC 3327 – introduces the PATH header field, used in conjunction with REGISTER requests and responses to REGISTER, to accumulate and transmit the list of proxies that have to be transited by inbound requests sent to the UA.
- RFC 3581 – introduces a new ‘rport’ parameter field for the VIA header field that allows a UA to request that the server send the response back to the source IP address and port from where the request originated. It is an essential part of getting SIP through NAT
- RFC 3326 – defines the ‘Reason’ header field. It provides the reason for initiating the request or to include a final status code in provisional responses.
- RFC 4412 and RFC 4411 – RFC 4412 introduces two new header fields “resource-priority” and “accept-resource-priority” to indicate request for priority treatment during emergencies. RFC 4411 defines an extension to the REASON header to be included in the BYE requests to allow a UA to know that it’s session is torn down (preempted) to allow a higher precedence session.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.6.4 SIP Infrastructure Extensions (General Uses)

These extensions to SIP, SDP and MIME are general purpose extensions introduced for various applications.

2.6.4.1 Fundamental Protocol Related

- draft-ietf-mmusic-ice-13 – defines a method for NAT traversal of media sessions for protocols using offer/answer model, like SIP media streams.
- RFC 3262 – in addition to final responses, SIP defined provisional responses (100-199) for informational purposes only. These are not sent reliably. RFC 3262 defines the Provisional Response ACKnowledgement (PRACK) method and an option tag (100rel) to provide reliable provisional responses.
- RFC 4028 – defines a keep-alive mechanism for SIP sessions by requiring the UAs to send periodic re-INVITE or UPDATE requests. This helps the Stateful Proxies keep the call-state status of all sessions current even in the event of improper terminations.
- RFC 4168 – specifies a mechanism of using SCTP as a transport protocol between SIP entities.

2.6.4.2 SDP Related

- RFC 4145 – defines an extension to SDP for setting up TCP based media sessions between UAs.
- RFC 4091 – defines a mechanism for including alternative types of network addresses (e.g., both IPv4 and IPv6) in SDP for a specific media session.

2.6.4.3 SIP Requests Related

- RFC 2976 – introduces the INFO message to carry mid-call application level signaling information along the session signaling path.
- RFC 3311 – defines a new UPDATE method for SIP to update session parameters (e.g., media stream characteristics) codecs, during “early media” or before the initial INVITE has been answered.

2.6.4.4 SIP Header Related

- RFC 3323 – defines the Privacy Header field for SIP. This RFC defines the roles and messages to be used by UAs and Intermediary Servers when the users choose not to divulge personal identity information.
- RFC 3841 – defines three new request header fields - Accept-Contact, Reject-Contact, and Request-Disposition. The Request-Disposition header allows UAs to express preferences as to how their requests are handled by the servers and the Accept-Contact and Reject-Contact headers allow the UAs to express a preferred feature set for target UAs.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- RFC 4244 – introduces a new header field, “History-Info”, to capture the history of requests from a UA that arrive at a particular application server or user.
- RFC 3420 – introduces a new header field “Service-Route”, that records a path of proxies and is included by a registrar server in its response to a REGISTER request from a UA. The UA then could use this service-route when requesting service through the specific service proxy.

2.6.5 SIP Limitations

SIP was designed to solve only a small set of problems and to allow interoperability with a broad spectrum of existing and future IP telephony protocols. To this end SIP provides four basic functions:

- User Location: mapping a user's name to their current network address (similar to DNS)
- Feature Negotiation: Allows User Agents (UA) to negotiate a common set of features
- Call participant management: adding, dropping, or transferring participants
- Modifying session features while a call is in progress.

Any other functions must be performed by other protocols. Its simplicity means that SIP is not a Session Description Protocol (SDP) nor is it able to perform Conference Control functions. It is also not a Resource Reservation Protocol (RSVP) and it has nothing to do with guaranteeing quality of service (QoS), e.g., IEEE 802.1p, Type of Service (ToS). SIP can work within a framework with other protocols to insure these roles are played out - but SIP does not perform these functions itself. SIP is regularly deployed alongside SOAP, HTTP, XML, VXML, WSDL, UDDI, SDP, RTP and a variety of other protocols.

Because of the simple nature of SIP many of the functions described in this report are not achievable with native SIP alone. Standard methods for deploying these features have not yet been ratified due to the myriad different potential methods to deploy any given feature. Vendors have independently chosen varying methods to solve these issues, which, in turn create a non-interoperable or proprietary situation. Due to this current situation where vendors have not reached agreement and the lack of standards it will be many years, if ever, before multi-vendor solutions with “plug-and-play” advanced features are available to the marketplace. This leads to the current environment, where COTS SIP products cannot be guaranteed to interoperate or even have similar feature sets.

2.6.6 Basic Flows

There are a few SIP transactions that must always occur in any SIP interaction. The examples below show state diagrams of SIP flows for User Registration and a basic phone



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

call between two users (UA). The flows discussed below are described in RFC3665 *SIP Basic Call Flow Examples*.

2.6.6.1 SIP Client Registration

Client registration is not absolutely necessary in order to complete a session (see Figure 2-1). If User A knows the network address of User B it can simply call it directly.

However, in a network with large groups of users this becomes unfeasible. In these cases a registration server is required. Registration either validates or invalidates a SIP client for user services provided by a SIP server. Additionally, the client provides one or more contact locations to the SIP server with the registration request. Registration is used by a Proxy to route incoming calls in an IP Telephony network. Registration is shown with authentication in these call flows. If authentication is not used, an imposter could *hijack* someone else's calls.

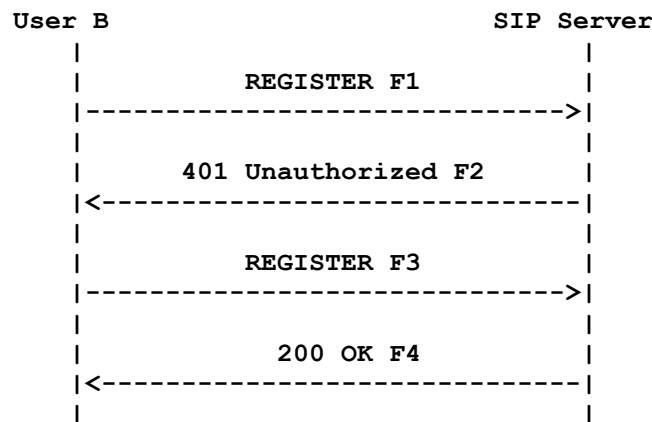


Figure 2-1 SIP Client Registration

User B initiates a new SIP session with the SIP Server, i.e., the user "logs on to" the SIP server. User B sends a SIP REGISTER request to the SIP server. The request includes the user's contact list. Contact list refers to the *Contact* field in the REGISTER header and contains all the caller's potential URL's. The SIP server provides a challenge to User B. User B enters her/his valid user ID and password. User B's SIP client encrypts the user information according to the challenge issued by the SIP server and sends the response to the SIP server. The SIP server validates the user's credentials. It registers the user in its contact database and returns a response (200 OK) to User B's SIP client. The response includes the user's current contact list in Contact headers. The format of the authentication shown is SIP digest as described by RFC2543. It is assumed that User B has not previously registered with this Server (see Figure 2-2).

Assuming User B's name is Bob, the SIP message details are as follows:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

F1 REGISTER Bob -> SIP Server

```
REGISTER sips:ss2.biloxi.example.com SIP/2.0
Via: SIP/2.0/TLS client.biloxi.example.com:5061;branch=z9hG4bKnashds7
Max-Forwards: 70
From: Bob <sips:bob@biloxi.example.com>;tag=a73kszlfl
To: Bob <sips:bob@biloxi.example.com>
Call-ID: 1j9FpLxk3uxtm8tn@biloxi.example.com
CSeq: 1 REGISTER
Contact: <sips:bob@client.biloxi.example.com>
Content-Length: 0
```

F2 401 Unauthorized SIP Server -> Bob

```
SIP/2.0 401 Unauthorized
Via: SIP/2.0/TLS client.biloxi.example.com:5061;branch=z9hG4bKnashds7
;received=192.0.2.201
From: Bob <sips:bob@biloxi.example.com>;tag=a73kszlfl
To: Bob <sips:bob@biloxi.example.com>;tag=1410948204
Call-ID: 1j9FpLxk3uxtm8tn@biloxi.example.com
CSeq: 1 REGISTER
WWW-Authenticate: Digest realm="atlanta.example.com", qop="auth",
    nonce="ea9c8e88df84f1cec4341ae6cbe5a359",
    opaque="", stale=FALSE, algorithm=MD5
Content-Length: 0
```

F3 REGISTER Bob -> SIP Server

```
REGISTER sips:ss2.biloxi.example.com SIP/2.0
Via: SIP/2.0/TLS client.biloxi.example.com:5061;branch=z9hG4bKnashd92
Max-Forwards: 70
From: Bob <sips:bob@biloxi.example.com>;tag=ja743ks76zflfH
To: Bob <sips:bob@biloxi.example.com>
Call-ID: 1j9FpLxk3uxtm8tn@biloxi.example.com
CSeq: 2 REGISTER
Contact: <sips:bob@client.biloxi.example.com>
Authorization: Digest username="bob", realm="atlanta.example.com"
    nonce="ea9c8e88df84f1cec4341ae6cbe5a359", opaque="",
    uri="sips:ss2.biloxi.example.com",
    response="dfe56131d1958046689d83306477ecc"
Content-Length: 0
```

F4 200 OK SIP Server -> Bob

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS client.biloxi.example.com:5061;branch=z9hG4bKnashd92
;received=192.0.2.201
From: Bob <sips:bob@biloxi.example.com>;tag=ja743ks76zflfH
To: Bob <sips:bob@biloxi.example.com>;tag=37GkEhw16
Call-ID: 1j9FpLxk3uxtm8tn@biloxi.example.com
CSeq: 2 REGISTER
Contact: <sips:bob@client.biloxi.example.com>;expires=3600
Content-Length: 0
```

Figure 2-2 Example of Registration



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.6.7 Basic SIP Call

SIP is used to provide signaling for a call between two users. RTP (Real Time Protocol) is then used to carry the voice payload after the call is setup. SIP is then used to tear down (hang-up) the call (see Figure 2-3).

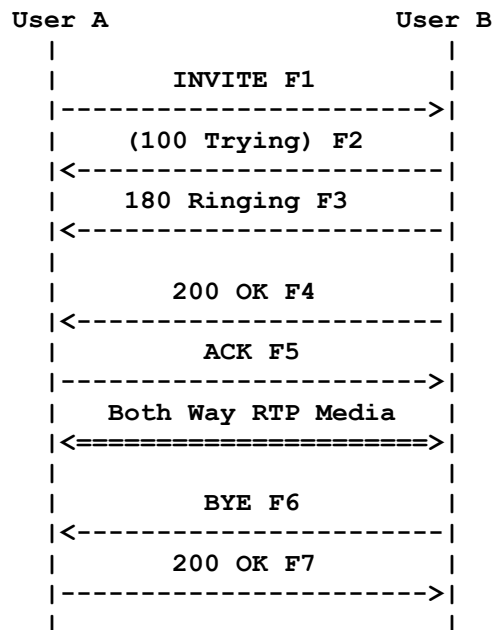


Figure 2-3 Basic SIP Call

In this scenario, User A completes a call to User B directly. User A sends an **INVITE** message to User B. This message indicates to User B that User A would like to set up a dialog. User B sends back a **100 TRYING** to acknowledge the **INVITE** request followed by a **180 RINGING** message to provide a ring-back tone to User A. Once User B decides to accept the call it returns a **200 OK** message to User A, who in turn accepts the call by responding with an **ACK** message. The call is now setup and RTP is used to carry the actual conversation or payload. Once the conversation is concluded, User B terminates the call by sending a **BYE** to User A, who in turn responds with a **200 OK** message.

Assuming User A's name is Alice and B's name is Bob, the SIP message details are as follows (see Figure 2-4):

F1 INVITE Alice -> Bob

INVITE sip:bob@biloxi.example.com SIP/2.0



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 INVITE
Contact: <sip:alice@client.atlanta.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=alice 2890844526 2890844526 IN IP4 client.atlanta.example.com
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F2 180 Ringing Bob -> Alice

SIP/2.0 180 Ringing
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>;tag=8321234356
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 INVITE
Contact: <sip:bob@client.biloxi.example.com;transport=tcp>
Content-Length: 0

F3 200 OK Bob -> Alice

SIP/2.0 200 OK
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>;tag=8321234356
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 INVITE
Contact: <sip:bob@client.biloxi.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: 147

v=0
o=bob 2890844527 2890844527 IN IP4 client.biloxi.example.com
s=-
c=IN IP4 192.0.2.201
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
F4 ACK Alice -> Bob

ACK sip:bob@client.biloxi.example.com SIP/2.0



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bd5
Max-Forwards: 70
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>;tag=8321234356
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 ACK
Content-Length: 0

/* RTP streams are established between Alice and Bob */

/* Bob Hangs Up with Alice. Note that the CSeq is NOT 2, since
   Alice and Bob maintain their own independent CSeq counts.
   (The INVITE was request 1 generated by Alice, and the BYE is
   request 1 generated by Bob) */

F5 BYE Bob -> Alice

BYE sip:alice@client.atlanta.example.com SIP/2.0
Via: SIP/2.0/TCP client.biloxi.example.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
From: Bob <sip:bob@biloxi.example.com>;tag=8321234356
To: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 BYE
Content-Length: 0

F6 200 OK Alice -> Bob

SIP/2.0 200 OK
Via: SIP/2.0/TCP client.biloxi.example.com:5060;branch=z9hG4bKnashds7
;received=192.0.2.201
From: Bob <sip:bob@biloxi.example.com>;tag=8321234356
To: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 BYE
Content-Length: 0
```

Figure 2-4 SIP Message Details

2.6.8 Implementation of Selected Features

Below we examine a set of selected features and the feasibility of implementing them using native SIP. Where the feature is achievable in native SIP, state flow diagrams and explanations are provided. Where they cannot be implemented in native SIP, an explanation is provided along with potential workarounds and/or problems that may arise if implemented in SIP.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2.6.8.1 Conference Call

A conference call is the creation of a group of end devices coupled together so all participants can hold a conversation. It requires action of the end user to pick up their end device or call into a conference number.

SIP was defined to allow for the establishment, maintenance, and termination of calls between one or more users. However, despite its origins as a large-scale multiparty conferencing protocol, SIP is used today primarily for point-to-point calls. This configuration is the focus of the SIP specification and most of its extensions. As a result, there is a lot of confusion about how SIP supports multi-party conferencing.

There are a number of conferencing models supported by native SIP. These models vary in range from Three-Party Calling with end system mixing, to large multicast conferences, to dial-in or dial-out conference servers, to ad-hoc centralized conferences, and to conferences using centralized signaling and distributed media. Most conference calls involving more than a dozen or so participants require more advanced features such as the ability of the moderator to mute all phones, set the length of time for the conference call, record and display the participants of the conference call, etc. However, there is no ratified standard for any of these models as of yet. A draft exists by Rosenberg, titled *draft-rosenberg-sip-conferencing-models-00* which describes all of these models in detail.

Standard native SIP cannot provide conferencing capabilities without the help of some form of SIP Proxy or IP-PBX for mixing the voice signals. The simplest example of conferencing is accomplished by first initiating a basic call between two users (A and B) as described in the previous section above. Once the call is in progress, B initiates a second, separate call to User C. Once the second call is in progress, B will then mix the streams and send B+C to A, and send A+B to C. In this example, User B is acting as the mixer/IP-PBX. This method of conference calling is how many IM (Instant Messaging) clients function. The number of participants is usually limited to a few participants. A separate server or PBX could just as easily perform the *mixer* function without affecting the SIP flow in any significant manner.

Changes Needed to SIP Protocol

No changes or modifications of SIP are required to support these feature, however, a User Agent Server (UAS) or mixer is required.

State Diagrams

The example below (see Figure 2-5) describes the simplest form of conference calling with SIP. All other models are variations of the same.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

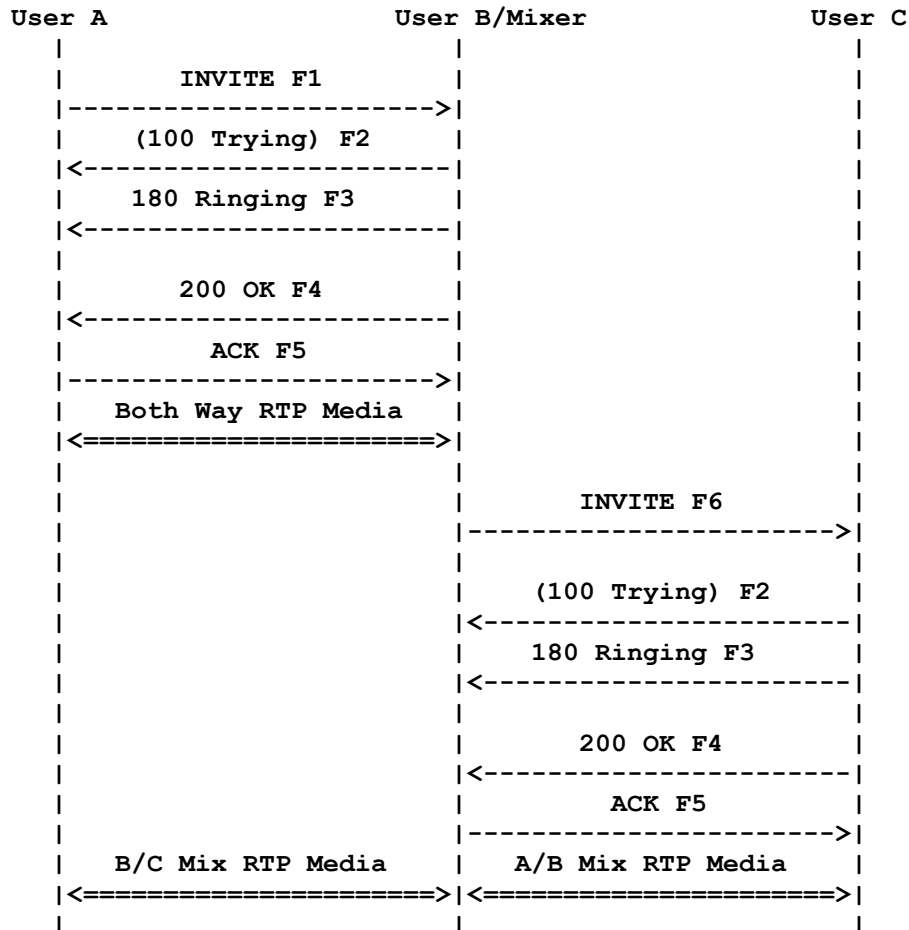


Figure 2-5 Simple Conference Call With SIP

See previous example for descriptions of SIP message details. The next example (see Figure 2-6) assumes that users A, B and C are already in conference using a proxy. The example describes how User D would be added to the conference. Only one addition is shown for the sake of simplicity. It should be remembered that any other additions would follow the same flow.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

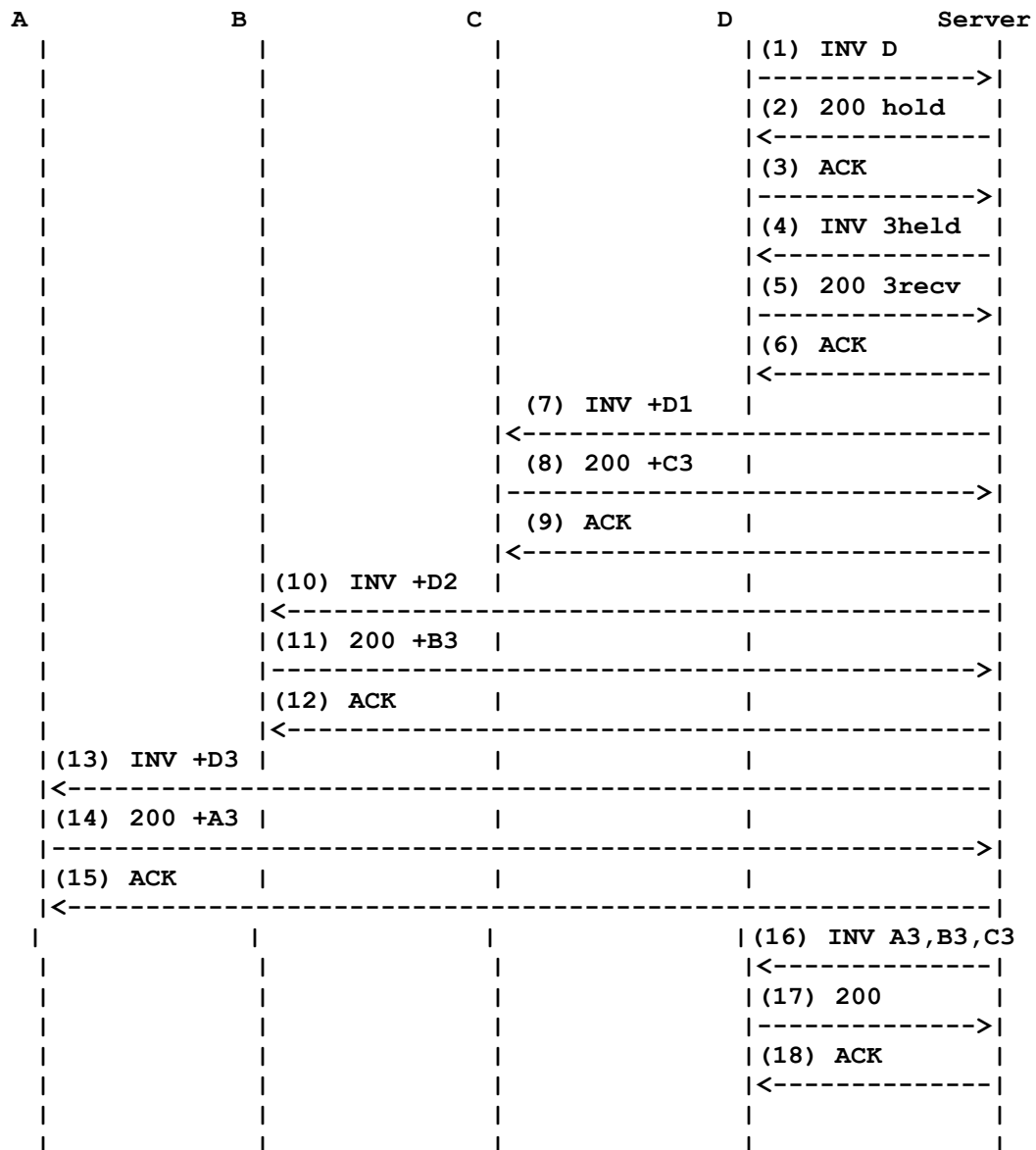


Figure 2-6 Example Conference Using a Proxy

As described in draft-rosenberg-sip-conferencing-models-00, “users joining “ is easily done. The participant that wishes to join simply sends an INVITE to the conference server, with the conference ID in the request URI (1). The conference ID (which is a SIP URL) can be learned by any number of means, including having it on a web page, receiving it in an email, etc.

For example, if D wishes to join sip:conference34@servers.com, D would send the following request:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
INVITE sip:conference34@servers.com
From: sip:D@example.com
To: sip:conference34@servers.com
```

2.6.8.2 Intercom

This feature involves the creation of a group of end devices coupled together so all participants can hold a conversation. It doesn't require the end user to pick up their end device or hang it up. The end device goes off-hook automatically and then returns to on-hook after the intercom is completed.

The Session Initiation Protocol does not currently provide a mechanism to force the UA (User Agent) to go *off-hook*. A UA could be configured to *auto-answer* incoming calls. However, this method has some security implications. In particular, there may be problems with potentially having an open-microphone when auto-answering a call. Other parties within the vicinity of the end device may over hear conversations they are not meant to. It could be an appropriate method for one-way announcements in some circumstances.

It is possible, and some manufacturers have implemented this method, to add a field to the standard INVITE header that would cause the receiving UA to go off-hook automatically with or without mute. In essence, the called device must understand this field or flag and it must be programmed to act on it. A device that does not support this field will simply ignore the flag and continue with normal operations. A number of manufacturers have successfully implemented this feature. Some with the method described here but most have done it with proprietary methods such as using proprietary protocols running over SIP. It goes without saying that until this method is ratified in an IETF RFC, there is no way to guarantee that any given SIP UA will support this feature. There is an Internet draft called "draft-ietf-sip-answermode-00" but it expired in June of 2006 with no action taken.

The creation of a group of end devices so all participants can hold a conversation is in essence a form of Conference Call. There are multiple methods of Conference Calling but two major types: dial-in and invite-based. A dial-in conference is one in which users must explicitly request to join the group. In an invite-based Conference, however, a user is invited (using the INVITE message) to join the group. The specific requirement here would be a basic invite-based conference and can be implemented as described above in the Conference Call section.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Changes Needed to SIP Protocol

Methods to extend SIP have been implemented by some manufacturers but an Internet draft must be proposed, discussed and ratified in order to provide this function in a standard manner and guarantee availability in any given SIP device.

State Diagrams

The state diagram would be identical to a basic call. A minor variation in the SIP message detail would be the only change. No State diagram is provided.

2.6.8.3 Group Page

The creation of a group of end devices coupled together to allow a one-way announcement can be done. The end device goes off-hook automatically and then returns to on-hook after the announcement is completed.

This feature has similar issues to the Intercom feature described above in that it involves calling a group of users simultaneously. Like the Intercom feature above, it would also be an invite-based conference call. The basic difference between these features is that the Intercom feature calls for a two-way conversation and Group Page calls for a one-way conversation but the mechanics are identical.

Changes Needed to SIP Protocol

An auto-mute function must be added to SIP.

State Diagrams

This feature has a state diagram flow identical to a conference call where a user is invited in rather than calling-in. See state diagram for Conference feature above. No state diagram is provided.

2.6.8.4 Priority Calls

The ability for an individual to break into a call that is in progress by use of a feature code and supervisor login. In Avaya's documentation this is called "Busy Verify".

SIP, as a signaling protocol, does not have the ability to break into ongoing calls. The problem is similar to the auto-answer issue described above.

Changes Needed to SIP Protocol

Extensions to SIP would need to be drafted and ratified in order to provide this function. The SIP protocol would need to have the ability to signal the UA and have the UA duplicate the incoming and outgoing voice streams and mirror them to UA, SIP Proxy, IP-PBX.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

State Diagrams

Not possible within the framework of SIP. No State diagram is provided.

2.6.8.5 NCS Voice Precedence System

Also known as Multi-Level Precedence and Preemption or MLPP gives individuals the ability to override a call that is in progress between two or more other parties. This feature is presently not being used onboard US Navy vessels. Instead, Priority Calling is being used.

Priority Calling differs from MLPP in that it is a method of inserting a third-party into an ongoing call without notification to the original parties. Also known as Busy Verify (Avaya) or Barge-In (Alcatel), it was traditionally used to check the status of a line by an operator or similar person. MLPP on the other hand, is a priority-based call override system. This system may affect all SIP elements in a network. However, this paper will only discuss it within the context of the end-user point of view.

MLPP notifies the called party with a tone prior to overriding the existing call. For example:

- Users A and B are having a conversation
- User C must communicate with A
- User C calls A using a priority level higher than the ongoing call between A and B
- Both A and B are presented with a tone warning them of an incoming higher priority call
- The call between A and B is dropped
- A new session between A and C is immediately started

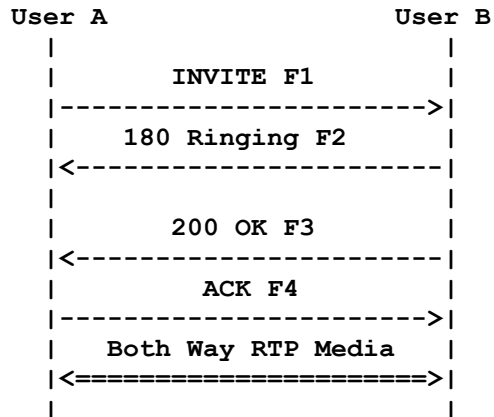
This feature is a SIP standard and is defined in RFC4412. The RFC adds two new fields to standard SIP messages. These fields can potentially appear in all types of SIP messages. It is mandatory only for handful of messages including INVITE, REFER, UPDATE, PRACK and ACK but optional in others.

No changes needed to SIP. At a minimum, support of RFC4412 is required by the UA's. Preferably, all SIP elements should support this RFC. The state flow for this feature looks identical to a basic call with the addition of the *Resource-Priority* field in the INVITE message (see Figure 2-7).



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



The relevant message headers are:

F1 INVITE User A -> User B

```

INVITE sip:UserB@biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: BigGuy <sip:UserA@atlanta.example.com>;tag=9fxced76s1
To: LittleGuy <sip:UserB@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 INVITE
Resource-Priority: dsn.flash
Contact: <sip:UserA@client.atlanta.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: ...
...
  
```

F2 180 Ringing User B -> User A

```

SIP/2.0 180 Ringing
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: BigGuy <sip:UserA@atlanta.example.com>;tag=9fxced76s1
To: LittleGuy <sip:UserB@biloxi.example.com>;tag=8321234356
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 INVITE
Contact: <sip:UserB@client.biloxi.example.com;transport=tcp>
Content-Length: 0
  
```

F3 200 OK User B -> User A

```

SIP/2.0 200 OK
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: BigGuy <sip:UserA@atlanta.example.com>;tag=9fxced76s1
To: LittleGuy <sip:UserB@biloxi.example.com>;tag=8321234356
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 INVITE
  
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
Contact: <sip:UserB@client.biloxi.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: ...

...
```

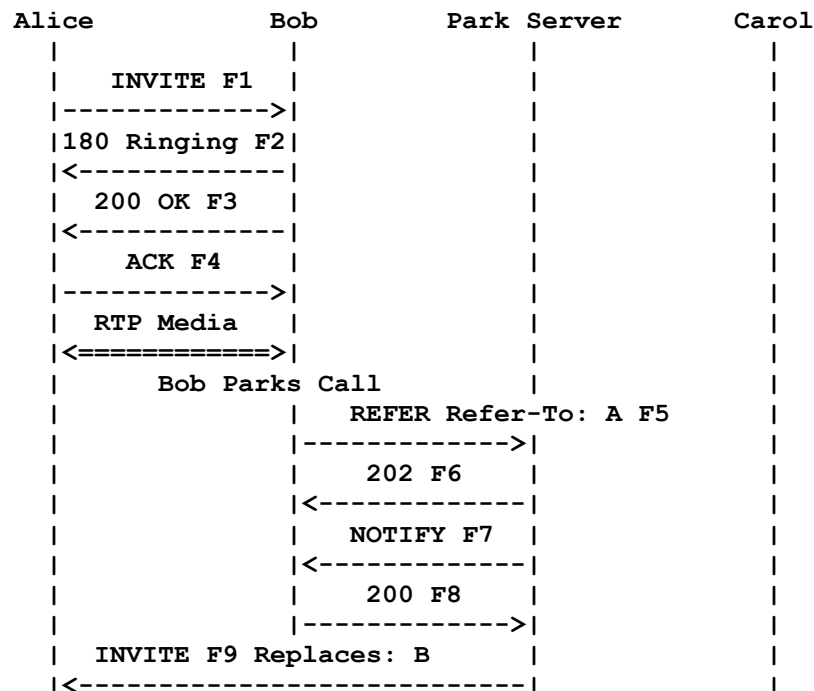
Figure 2-7 Invite Message

2.6.8.6 Call Park

The ability to park a call onto a virtual extension and then have a third-party or the same individual, retrieve the call. To the caller, the call appears to be on hold and is presented with “music on hold”.

This feature, like many others, can be implemented with SIP in several ways. It is defined RFC 4240 and in the *draft-procter-sipping-call-park-extension-00* draft document. Several different methods are also described in books and SIP-related sites on the Internet. For this reason, this paper will describe the most basic and simple version of these methods. The method described below appears in *draft-procter-sipping-call-park-extension-00* and a similar method is described in *draft-ietf-sipping-service-examples-06*. The only requirement is a “Park Server”. Other methods may require modifications and/or additional servers.

This method adds an *orbit* tag to the REFER message. This tag is user defined provides a basic extension that can be used later to retrieve the call. It provides a way to avoid two callers being parked simultaneously in the same place (see Figure 2-8).





Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

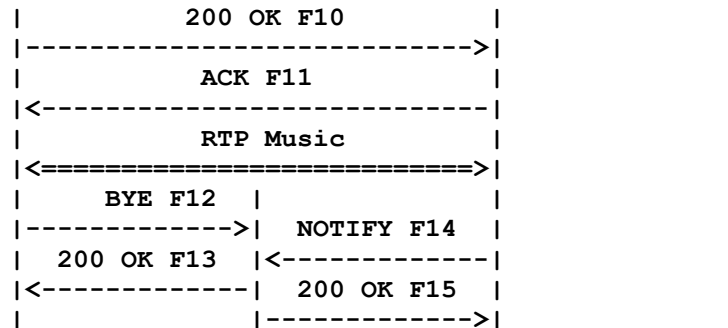


Figure 2-8 Call Park

The URI <sips:park@server.example.com;orbit=1234> is used instead of directing the request to the URI <sips:park@server.example.com>. The addition of the orbit parameter effectively tags the parked call with a short memorable code entered by the user (see Figure 2-9 and Figure 2-10).

F5 REFER Bob -> Park Server

```

REFER sips:park@server.example.com;orbit=1234 SIP/2.0
Via: SIP/2.0/TLS client.biloxi.example.com:5061;branch=z9hG4bKnashds9
Max-Forwards: 70
From: Bob <sips:bob@biloxi.example.com>;tag=02134
To: Park Server <sips:park@server.example.com;orbit=1234>
Call-ID: 4802029847@biloxi.example.com
CSeq: 1 REFER
Refer-To: <sips:alice@client.atlanta.example.com?Replaces=
12345601%40atlanta.example.com%3Bfrom-tag%3D314159%3Bto-tag%3D1234567>
Referred-By: <sips:bob@biloxi.example.com>
Contact: <sips:bob@client.biloxi.example.com>
Content-Length: 0
  
```

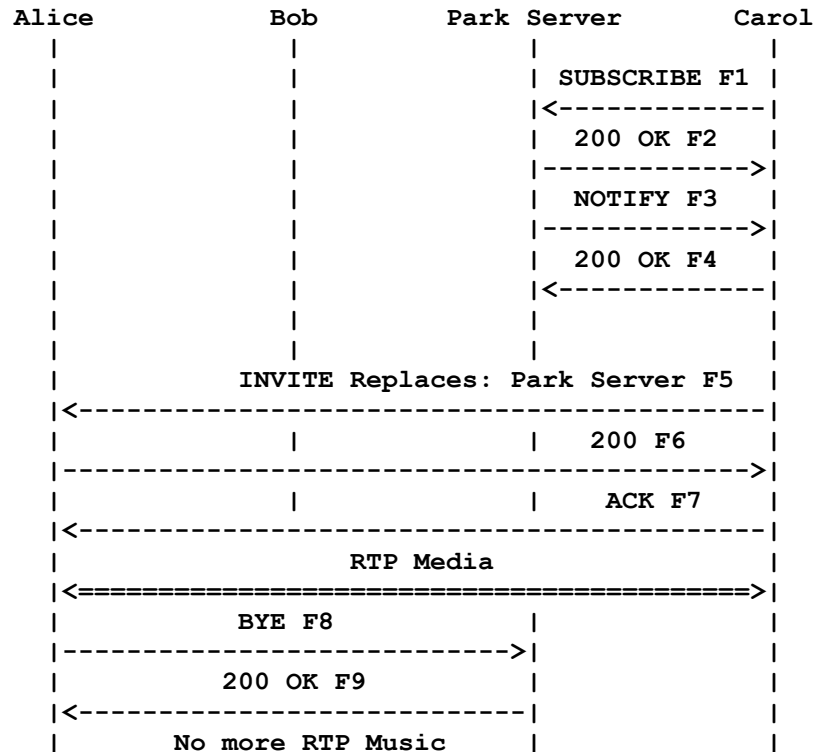
Figure 2-9 Park Server 1

Alice is now *parked* at the Park Server. In order to retrieve the call, Carol calls the Park Server. The Server notifies Carol of the URI required to retrieve the call in the NOTIFY message. Remember that the URI contains the *orbit* tag.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



F1 SUBSCRIBE Carol -> Park Server

```
SUBSCRIBE sips:park@server.example.com;orbit=1234 SIP/2.0
Via: SIP/2.0/TLS chicago.example.com:5061;branch=z9hG4bK92bz
Max-Forwards: 70
From: Carol <sips:carol@chicago.example.com>;tag=8672349
To: <sips:park@server.example.com;orbit=1234>
Call-ID: xt4653gs2ham@chicago.example.com
CSeq: 1 SUBSCRIBE
Contact: <sips:carol@client.chicago.example.com>
Event: dialog
Subscription-State: active;expires=0
Accept: application/dialog-info+xml
Content-Length: 0
```

F2 200 OK Park Server -> Carol

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS chicago.example.com:5061;branch=z9hG4bK92bz
;received=192.0.2.114
Max-Forwards: 70
From: Carol <sips:carol@chicago.example.com>;tag=8672349
To: <sips:park@server.example.com;orbit=1234>;tag=1234567
Call-ID: xt4653gs2ham@chicago.example.com
CSeq: 1 SUBSCRIBE
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Content-Length: 0

F3 NOTIFY Park Server -> Carol

```
NOTIFY sips:carol@client.chicago.example.com SIP/2.0
Via: SIP/2.0/TLS chicago.example.com:5061;branch=z9hG4bK93ca
Max-Forwards: 70
To: Carol <sips:carol@chicago.example.com>;tag=8672349
From: <sips:park@server.example.com;orbit=1234>;tag=1234567
Call-ID: xt4653gs2ham@chicago.example.com
CSeq: 2 NOTIFY
Contact: <sips:park@server.example.com;orbit=1234>
Event: dialog
Subscription-State: terminated
Content-Type: application/dialog-info+xml
Content-Length: ...
```

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  version="0" state="full"
  entity="sips:park@park.server.example.com;orbit=1234">
  <dialog id="94992014524" call-id="12345600@atlanta.example.com"
    local-tag="3145678" remote-tag="1234567" direction="recipient"
    remote-uri="alice@atlanta.example.com"
    remote-target="alice@client.atlanta.example.com">
    <state>confirmed</state>
  </dialog>
</dialog-info>
```

F4 200 OK Carol -> Park Server

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS chicago.example.com:5061;branch=z9hG4bK93ca
To: Carol <sips:carol@chicago.example.com>;tag=8672349
From: <sips:park@server.example.com;orbit=1234>;tag=1234567
Call-ID: xt4653gs2ham@chicago.example.com
CSeq: 2 NOTIFY
Contact: <sips:carol@client.chicago.example.com>
Content-Length: 0
```

Figure 2-10 Park Server 2

2.6.8.7 Directed Call Pick-up

This feature provides the ability to pick-up a call that is ringing on another end device. Call Pickup is described in the IETF draft called *draft-worley-sipping-pickup-02*. This document states:

“There are several different schemes for implementing call pickup. The basic method is the one specified in the Sylantro "SIP-B" specification, which despite its proprietary air,



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

uses standard SIP features in an end-point call control (EPCC) style. All other methods are variations on the same theme, usually by using an agent process (in a proxy or communications server) to provide a feature that the user agents are lacking.

Like call transfer, effecting call pickup requires some support from the caller's end. These caller-end features will, therefore, soon come to be considered necessary for any "quality" SIP implementation."

As there are so many variations and possible implementation methods (none of them standard) for this feature, this paper will focus on the simplest of them and refer the reader to *draft-worley-sipping-pickup-02*, *draft-ietf-sipping-service-examples*, and *draft-procter-sipping-call-park-extension* for more in-depth discussion of the more complex methods.

The basic method involves calling the ringing extension and polling it for status of its current calls. The ringing extension responds with, among other things, the caller URI. The calling UA can then be sent an INVITE thereby "picking-up" the call.

The basic SIP-B pickup sequences are as follows (only principal messages are shown). Assume the incoming call is to the callee phone, extension 123, and the phone executing the pickup is extension 456 (see Figure 2-11):



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

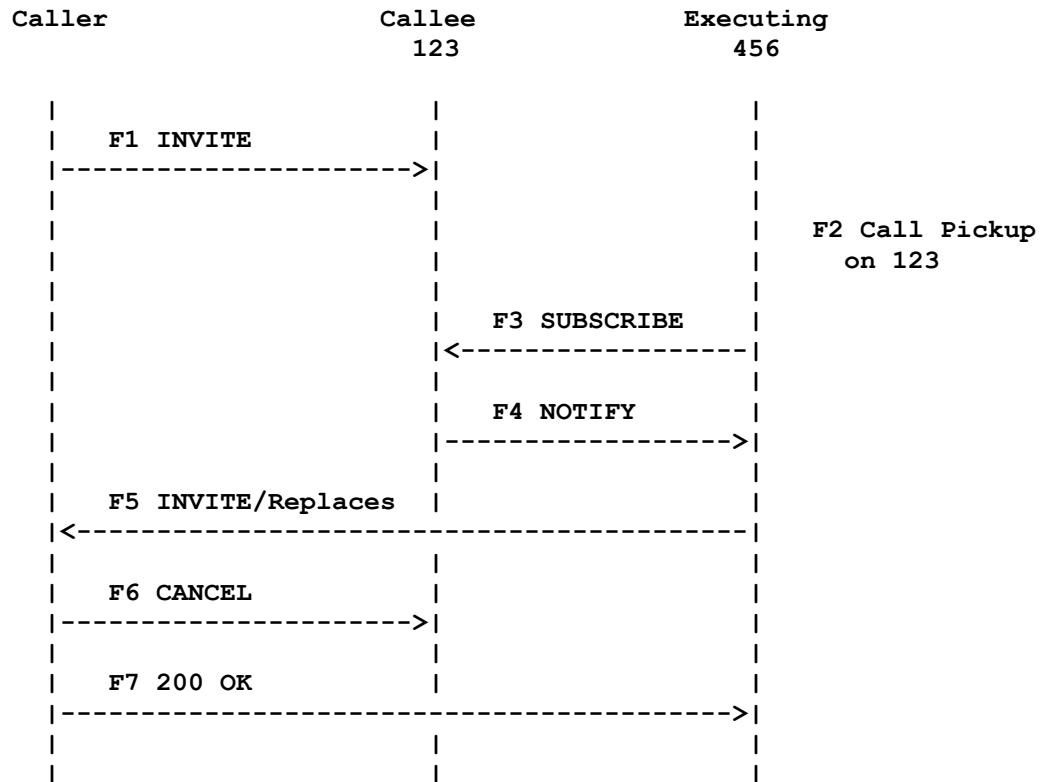


Figure 2-11 Directed Call Pick-up

F1 - The caller (AOR Caller@example.com) sends an INVITE with URI 123@example.com to phone 123.

F2 - The user of phone 456 activates the call pickup feature for extension 123.

F3 - Phone 456 sends a SUBSCRIBE with URI 123@example.com to phone 123, requesting "Event: dialog" and "Expires: 0".

F4 - Phone 123 sends one NOTIFY to phone 456 giving the status of its current dialogs for AOR 123@example.com, which includes the early dialog of INVITE 1 from Caller, and gives the "remote identity" and "remote target" of the dialog (which are the From: and Contact: of INVITE 1), one of which is "Caller@example.com".

F5 - Phone 456 sends INVITE 5 to Caller@example.com. It has a Replaces: header specifying the dialog parameters sent in the NOTIFY. The Replaces: header contains the "early-only" option, so that the pickup operation fails if extension 123 answers the call. As a consequence of executing the INVITE/Replaces, the caller sends a CANCEL of its INVITE 1 to phone 123.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The caller sends a 200 response to the INVITE 5 from phone 456. At this point, Caller is talking to phone 456.

2.6.8.8 Group Call Pick-up

The SIP flows for this function are identical to the flows required for Directed Call-Pickup (above). The exception being that the called extension is a conference URI. In other words, the group of end-devices must be in a conference. The user may then poll the conference URI to obtain the ringing party's URI.

2.6.8.9 Recording of Calls

This feature would provide the ability to record calls from a set of pre-defined end devices as soon as they are off hook. This would be used for all calls that the bridge handles.

There are a number of ways in which this feature might be implemented. The community has discussed this subject extensively since the year 2000. Several ideas have been bandied about but no drafts or standards have come of it. The potential implementations would all require the creation of a Conference Call and the automatic addition (or INVITE) of a *recorder*. The *recorder* would simply act as a standard UA and store the mixed streams it receives.

This can be achieved in a number of different ways but this paper will only describe the simplest of these methods, as no standard yet exists. Assume that the bridge UA is A, the *recorder* is B and the called/calling party is C. The simplest method would be to program the user's UA (A) to automatically INVITE the *recorder* (B) and conference it in along with C every time it goes off-hook.

The *recorder* device (B) could be implemented in any number of ways as well. It could be implemented in:

- The UA itself: allowing the placement of recorders anywhere they are needed.
- A "Black Box": this could be placed throughout the ship and be able to service multiple UA's simultaneously and provide redundancy in case of failure.
- A PBX: in this manner, the PBX could be programmed with all the extensions requiring this service.

Changes Needed to SIP Protocol

No changes to SIP need be implemented. However, a standard should be proposed and ratified that defines the methodology for achieving this functionality. Until a standard is



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

available there is no guarantee any two vendors will implement this feature in the same way.

State Diagrams

SIP state flows for this feature would be identical to that used in Conference Calling. See Conference Call section above for diagrams.

2.7 Summary

After much research, the salient point to be made is simple: all of the above features could be implemented with SIP. However, for any given feature there are myriad ways to achieve it. This is, coupled with SIP's simplicity, we believe, the reason for the current state of confusion with regards to SIP standardization. Some of these features can be implemented today with the standards that already exist. Others could be achieved in a proprietary manner, yet using SIP, through some creative thinking and development of private extensions to the protocol. This, however, creates a situation where only devices explicitly created to support said features would actually work.

In addition, we can envision that eventually the SIP protocol standard will be completed in such a manner as to provide standard methods of implementation for these and other telephony features. The real question is: when?



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

References

1. Das, K. P., Hubbart, J., and T. Rumland, *Intelligent Advanced Communications - Network Infrastructure*, April 2, 2007.
2. *The Opportunity: Pervasive Computing* - Indiana University
http://www.indiana.edu/~ovpit/ipcres/index_4.html, July 16, 2007
3. Recommendation H.323, International Telecommunications Union
<http://www.itu.int/rec/T-REC-H.323/e>, July 16, 2007
4. Session Initiation Protocol - Wikipedia, the free encyclopedia,
http://en.wikipedia.org/wiki/Session_Initiation_Protocol, July 16, 2007
5. RFC 3261, <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007
6. RFC 2543, <http://www.ietf.org/rfc/rfc2543.txt>, July 6, 2007



CHAPTER 3 IAC Security

3.1. Introduction

VoIP systems take a wide variety of forms, including traditional telephone handsets, conferencing units, and mobile units. In addition to end-user equipment, VoIP systems include a variety of other components, including call processors/call managers, gateways, routers, firewalls, and protocols. Most of these components have counterparts used in data networks, but the performance demands of VoIP mean that ordinary network software and hardware must be supplemented with special VoIP components. Not only does VoIP require higher performance than most data systems, the critical services such as Emergency 911, must be accommodated. One of the main sources of confusion for those new to VoIP is the (natural) assumption that because digitized voice travels in packets just like other data, existing network architectures and tools can be used without change. However, VoIP adds a number of complications to existing network technology, and these problems are magnified by security considerations [1]. VoIP has inherent weaknesses and is vulnerable at multiple points in the network infrastructure. VoIP must be secure in order to ensure the availability of the voice system, and to protect the content value and integrity of voice conversations.

3.2. Security

Security is probably the most important part of a converged network consisting of voice and data streams. Aside from the normal security issues revolving around e-mail, network infrastructure and virus scanning, security for the voice must be designed up front. There is no point to deploying a VoIP system without ensuring that the data network is as secure as possible. Adding VoIP to an existing installation requires a robust security analysis prior to making design decisions since security features to lock down the VoIP system can and will have an impact on data throughput.

3.2.1 Security Threats

Security threats are categorized into broad groups that define the type of threat they represent. The major threats to a VoIP network are:

- Theft of Service
- Unwanted Contact
- Denial of Service
- Impersonation
- Eavesdropping
- Interception



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Within each threat category, many variations of the threat exist. Theft of service applies to phone services that are stolen, where unwanted contact includes harassment and Spam Over Internet Telephony (SPIT). Denial of service is probably the most disruptive attack where the network resources and bandwidth are exhausted. Impersonation refers to false identities or to the rights any individual or group has been assigned. Eavesdropping refers to reading or gleaning information from the data packets and packet interception involves actual packet manipulation, packet rerouting and alteration.

Firewalls, gateways, and other such devices can help keep intruders from compromising a network. However, firewalls are no defense against an internal hacker. Additional layers of defense are necessary at the protocol level to protect the voice traffic. In VoIP, as in data networks, this can be accomplished by encrypting the packets at the IP level using IPSec, or at the application level with secure RTP, the real-time transport protocol (RFC 3550). However, several factors, including the expansion of packet size, ciphering latency, and a lack of QoS urgency in the cryptographic engine itself can cause an excessive amount of latency in the VoIP packet delivery. This leads to degraded voice quality. Careful network design decisions will greatly improve latency.

Remember that VoIP is data and is transmitted in digital packet form. This means that the voice transmissions can be now attacked, hacked, intercepted, manipulated, re-routed and degraded just as any data packet on the data network can. Viruses, worms, Trojan horses, denial of service attacks and hijacking are all possibilities on the VoIP network.

3.2.3 Security Tactics

Security must be implemented in a layered approach. This means that security has to involve the entire system. Each component must have a focus on security starting with the End Instrument (IP Phone) by hiding the phone/network parameters. Next the Call Servers, Media Gateways, Session Border Controllers and all Routers, Switches and Firewalls must all be locked down requiring administrative access for management changes and UserId/Password for use. Also, many of the data network precautions, such as virus scan software and an effective patch management system, need to be in place to keep the soft phones, servers and Personal Computers up to date. Next all voice streams and call signaling should be encrypted, ideally end-to-end. Voice should be encrypted with Secure RTP (SRTP) using 128-bit Advanced Encryption Standard (AES) Signaling should use Secure Socket Layer (SSL)/ Transport Layer Security (TLS) wherever possible or alternatively IPSec can be used to encrypt everything.

From a purely network approach; security can be applied by segregation. Each type of VoIP equipment type should be allocated to their own Virtual Local Area Networks (VLANs). VLANs are used to segment voice components and to segment the data network. Softphones that require access to both the voice and data VLANs should be allocated with care. Security studies from National Security Agency (NSA) and Defense



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Information Systems Agency (DISA) strongly suggest not using softphones or greatly limiting their use.

Physical network security with regards to critical network components, computer rooms, wiring closets, server rooms must have their access controlled at all times. Many network disruptions can be initiated when physical security of the VoIP components has been compromised.

3.2.4 Software Management

Software management and updating require secure access. The following list outlines some common management recommendations:

- Remote management should only be performed over encrypted connections.
- Proper password management techniques should be used.
- Any default passwords must be changed. Passwords need rotation.
- System actions should be logged with appropriate audit capabilities.
- Only secure connections should be used for web access, i.e., Secure Socket Layer (SSL)/ Hypertext Transfer Protocol Over Secure Socket Layer (HTTPS).
- Set software loads should be encrypted and tamper-proof.
- Network Service providers should run the minimum of services required.

Connection of a set to the system must require an initial authentication and authorization.” [2] To summarize, all VoIP traffic should be encrypted. There are multiple options including VPNs, SRTP (Secure RTP) and IPSec, but insuring that the selected encryption method is efficient and fast is a critical design issue. Otherwise, performance and throughput may be negatively impacted. Segment the network into VLANs that contain like equipment. For example, all softphones belong to their own VLAN as does the data Network. Segmentation provides another form of separation between voice and data traffic. The primary goal is to avoid or greatly reduce the commingling of voice and data traffic on the same network segment. The network must be actively monitored for unauthorized or non-compliant technologies supporting the VoIP network. This includes identifying devices with non-standard configurations. Make VoIP servers physically secure by adopting technologies such as firewalls and intrusion detection. Use firewalls that can handle the unique attributes of VoIP traffic. Require all users to login to access the VoIP network. A VoIP handset should be treated no differently than a user’s computer where network access is governed by login and password.

3.2.5 Security Policies

Security policies will differ for each installation. Reference [4] “Network Infrastructure, Security Technical Implementation Guide V6R4” is an excellent resource for guidelines



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

when deploying a security policy across the infrastructure. Reference [3] “IPT & VoIP STIG, V2R2 21 April 2006, Internet Protocol Telephony & Voice Over Internet Protocol, Security Technical Implementation Guide V2R2” provides specific security guidelines for VoIP installations. The Defense Information Systems Agency (DISA) developed references [3] and [4] for the Department of Defense (DOD). The National Security Agency’s (NSA) Systems and Network Attack Center (SNAC) have developed additional security configuration guidelines and checklists. Additional information may be obtained from the NSA website outlined in reference [5].

3.3. Government and Industry Contributions

The government and industry have contributed a number of documents for specifying and/or analyzing aspects of VoIP security. This section identifies some of the most relevant and widely used documents.

3.3.1 Defense Information Systems Agency

The Defense Information Systems Agency (DISA) maintains a set of documents, called Secure Technical Implementation Guides (STIGs), that detail how devices need to be designed and configured to meet DISA’s requirements for military departments to connect IT assets to the global DoD networks (specifically to networks connected to the Global Information Grid (GIG) and/or Defense Switched Network (DSN)/ Defense Red Switch Network (DRSN)). These requirements are the basis for DISA’s Information Assurance (IA) certification testing. Most devices need to conform to several applicable STIGs. For example, a server on a VoIP network may need to use the VoIP STIG[3], Enclave STIG[8], DSN STIG [6], UNIX STIG[9], etc. Each STIG has an accompanying checklist, which enumerates each requirement in the STIG, instructions to test compliance, and typical steps to remediate non-compliance. Some STIGs, such as the operating system STIGs and some database STIGs, include system readiness review scripts to automate the checklist process. Beyond the DoD, the STIGs can be used as best-practices guides.

3.3.2 National Institute of Standards and Technology

The National Institute of Standards and Technology (NIST) have published a few relevant documents. The special publications series SP 800-58 [10] is one of the most comprehensive documents on the subject of securing VoIP systems. It includes a description of security issues affecting VoIP and recommendations (as well as concerns) for deploying COTS VoIP solutions today. NIST is also responsible for the Federal Information Processing Standards (FIPS). FIPS 140-2[11] gives requirements for designing cryptographic modules suitable for protecting sensitive government data. FIPS 140-2 certification is a requirement for any product using encryption to be allowed onto a DoD network.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

NIST also plays a role, through the National Information Assurance Partnership, for Common Criteria (CC) evaluations. CC plays a role in Evaluated Assurance Level (EAL) certification, which is a fundamental step for vendors to sell information assurance products to the DoD. The CC framework itself is an ISO/IEC document [12]. It defines the process for building a Protection Profile (PP), which is a set of requirements for a functional device (e.g., a router or firewall). For a product to conform to the PP, the vendor must demonstrate that its product mitigates the threats outlined in the PP.

3.3.3 Other Relevant Contributors

The best-known requirements document for TDM-based telecommunications systems is Telcordia's GR-815 [13]. Its purpose was to provide requirements for securing the national telephone infrastructure in accordance with the Telecommunications Act of 1996. The document is primarily geared toward securing the core TDM switches (e.g., 5ESS® and tandem switches) and customer premises equipment.

The Alliance for Telecommunication's Industry's baseline security requirements [OAMP] is similar to GR-815, but focus on the management interfaces of IP-based systems (including but not specifically VoIP). Other industry and government specifications and requirements documents commonly cite these requirements.

Telephony requirements tend to use Chairman of the Joint Chiefs of Staff Instruction (CJCSI) instructions instead of DoD instructions. CJCS Instruction 6215.0.B[14] provides policies for DoD voice networks. CJCS Instructions 6510.01D[15] and 6211.02B[16] give requirements for DoD's global telecommunications networks (voice and data) and their interconnections including cross-domain solutions.



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

This page intentionally left blank.



CHAPTER 4 Open Source Feasibility

4.1 Introduction

The use of open source stacks had several reasons at the beginning of this phase, the primary reason being the investigation of different ways to implement AS-SIP into a SIP stack. The open source stacks allowed for several different styles of stacks to be evaluated and then reviewed for the pitfalls of adding AS-SIP to the stack. From this research the first question to be answered was: how should AS-SIP be added to the SIP stack source? Three methods of implementation were considered:

- Can the AS-SIP be added to the stack directly,
- Can AS-SIP be added as a “bolt-on” appendage,
- Or added as some combination of the two?

Which of the three could be implemented cleanly and still be supportable was the deciding factor. It was determined that the bolt-on appendage was not an option that worked cleanly since the stacks didn’t allow the information that was required to flow up from the stack to the supporting code. In the same manner, they did not allow for changes to the headers and parameters to be processed through the stack.

The modification of the stack directly made the open source no longer compatible with what was being working in the community. This was found to be the reason why sipXtapi 3.3.0 had been created, the problem was that it did not get a community that maintained it. This was an area that you do not want to create a open source project that there is no community to benefit and or contribute to it. This is where the oSIP was a great solution to start from since it was very open and was a great building block. The problem was that it would require to major investment on the parts that were not part of the project but were parts of the community.

The best solution was the sipXtapi 3.3.0 with the combined solution with changes to the stack as well as adding on functionality that had hooks into the stack that we added.

Three different open source SIP stacks were evaluated and implemented to varying degrees as potential candidates for our Assured Services SIP (AS-SIP) end instrument prototype. At a minimum, we wanted SIP stacks that included as much built in support as possible for the major UCR requirements that were not unique to AS-SIP. Finding stacks that included all of these common requirements such as Internet Protocol Version 6(IPV6), Secure Real Time Protocol(SRTP), Transport Layer Security(TLS), Provisional Response Acknowledgement (PRACK), and Internet Protocol Security(IPSEC) was not possible. We exhausted every



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

available SIP stack and/or complete soft phone before settling in on OSIP, sipXtapi and finally sipXTapi3.3 . This document will detail our findings with each of these 3 SIP stacks. All of the “complete” open source SIP phones we looked at were eliminated from consideration. Most did not include the entire source and or tools required to build the application or were too tightly coupled with their particular GUI display software.

4.2 SIP and Assured Services SIP

Understanding the differences between Assured Services SIP and standard SIP is not a trivial task. While the DOD Unified Capabilities Requirements documentation does an excellent job of identifying the various networking components that each requirement applies to (LSC,EI, MFSS, GEI, TA, IAD, MG etc...) there was no attempt made to highlight the actual changes made to the original IETF RFCs and standards that define AS-SIP. As it turns out, the SIP related UCR requirements are no different than the numerous RFCs and standards they apply to and in many cases are simply quoted word for word in the UCR.

RFC 5638 (Simple SIP) specifies the support of only 11 RFCs necessary to create a SIP appliance with presence, instant messaging, audio and video communications. The RFC blames "the emulation of telephony and its model of the intelligent network" as the reason there are now more than 100 RFCs that can define the behavior of a SIP appliance. Unfortunately for us, "emulating the existing telephony model" is exactly what the UCR requires us to do. The UCR requires support for nearly 200 RFCs although it should be noted that not all pertain directly to SIP. It is the substantially large number of requirements for the end instrument that make AS-SIP different from your typical SIP stack. This is not to say there are no variations from the original RFCs because there are plenty although most are subtle.

4.2.1 AS-SIP Messages

AS-SIP as defined by the DOD UCR document, is no different than standard SIP in terms of the overall structure and messaging. A brief summary of assured services SIP related requirements are as follows.

An AS-SIP message is the same as a standard SIP message and is either a request from a client to a server, or a response from a server to a client.

```
generic-message =  start-line
                  *message-header
                  CRLF
                  [ message-body ]
start-line = Request-Line / Status-Line
```

AS-SIP Requests

Request-Line = Method SP Request-URI SP SIP-Version CRLF

Required methods for AS-SIP:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

ACK	(RFC 3261)
BYE	(RFC 3261)
INVITE	(RFC 3261)
CANCEL	(RFC 3261)
REGISTER	(RFC 3261)
OPTIONS	(RFC 3261)
PRACK	(RFC 3262) AS-SIP req.
UPDATE	(RFC 3311) AS-SIP req.
REFER	(RFC 3515) AS-SIP req.
NOTIFY	(RFC 3265) AS-SIP req.

Example AS-SIP message:

INVITE sip:bob@biloxi.com SIP/2.0

A valid AS-SIP request formulated by a UAC MUST, at a minimum, contain the following header fields: “To”, “From”, “CSeq”, “Call-ID”, “Max-Forward” and ”Via” These above six header fields are the fundamental building blocks of all AS-SIP messages and are defined as follows.

To:

The To header field first and foremost specifies the desired "logical" recipient of the request. For AS-SIP, the userinfo part must contain the 10 digit DSN number plus the hostname. The To header field contains the address of record whose registration is to be created, queried, or modified. The To header field and the Request-URI field typically differ, as the former contains a user name. This address-of-record MUST be a SIP URI or SIPS URI. AS-SIP: requires "userinfo"

Example “To” headers:

To: The Operator <sip:operator@cs.columbia.edu>;tag=287447
t: sip:+12125551212@server.phone2net.com
(AS-SIP must contain the 10 digit number)

From:

The From header field contains the address-of-record of the person responsible for the Request (e.g. register). The value is the same as the To header field unless the request is a third-party registration for Registration.

Call-ID:

All registrations from a UAC SHOULD use the same Call-ID header field value for registrations sent to a particular registrar. If the same client were to use different Call-ID values, a registrar could not detect whether a delayed REGISTER request might have arrived out of order.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

CSeq:

The CSeq value guarantees proper ordering of REGISTER requests. A UA MUST increment the CSeq value by one for each REGISTER request with the same Call-ID.

Max-Forwards:

Serves to limit the number of hops a request can make on the way to its destination. It consists of an integer that is decremented by one at each hop.

Example: Max-Forwards: 70

Via:

The Via header field indicates the transport used for the transaction and identifies the location where the response is to be sent. A Via header field value is added only after the transport that will be used to reach the next hop has been selected. Copy from all requests to the response.

```
proto-name
 / proto-version
 / / transport protocol
 / / / host
 / / / / host port
 / / / / /
```

Example: Via: SIP/2.0/TLS atlanta.example.com:5060
v: SIP/2.0/TCP client.atlanta.example.com:5060
;branch=z9hG4bK74b76
;received=192.0.2.101

The “Request-URI” and “Contact” headers may also be required depending on the type of request:

Request-URI:

The Request-URI names the domain of the location service for which the registration is meant. The "userinfo" and "@" components of the SIP URI MUST NOT be present.

Example: "sip:chicago.com"

Contact:

REGISTER requests MUST contain a Contact header field with zero or more values containing address bindings.

AS-SIP Responses:

Status-Line = SIP-Version SP Status-Code SP Reason-Phrase CRLF



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Example: SIP/2.0 200 OK

SIP State

There are 2 types of client transaction state machines, depending on the method of the request passed by the TU.

- 1.) Client transactions for INVITE requests.
- 2.) Client transactions for all requests except INVITE and ACK or "non-INVITE client transactions"

Note that there is no client transaction for ACK. If the TU wishes to send an ACK, it passes one directly to the transport layer for transmission. The INVITE transaction is different from those of other methods because of the human input normally required to complete it and consequently its relatively long life time.

Support for the following SIP headers is required for AS-SIP:

Accept	(RFC 3261)
Alert-Info	(RFC 3261)
Allow	(RFC 3261)
Allow-Events (u)	(RFC 3265) AS-SIP req.
Authorization	(RFC 3261 generate and send only AS-SIP)
Call-ID (i)	(RFC 3261)
Contact (m)	(RFC 3261)
Content-Disposition	(RFC 3261)
Content-Length (l)	(RFC 3261)
Content-Type (c)	(RFC 3261)
CSeq	(RFC 3261)
Date	(RFC 3261)
Event (o)	(RFC 3265) AS-SIP req.
Expires	(RFC 3261)
From (f)	(RFC 3261)
Max-Forwards	(RFC 3261)
Min-Expires	(RFC 3261)
Min-SE	(RFC 4028) AS-SIP req.
P-Asserted-Identity	(RFC 3325 receive only) AS-SIP req.
Proxy-Authenticate	(RFC 3261 receive only)
Proxy-Authorization	(RFC 3261 generate and send only)
Proxy-Require	(RFC 3261 generate for Classified network only)
Rack	(RFC 3262) AS-SIP req.
Reason	(RFC 3326) AS-SIP req.
Record-Route	(RFC 3261)
Refer-To (r)	(RFC 3515, 4508) AS-SIP req.
Replaces	(RFC 3891) AS-SIP req.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Require	(RFC 3261)
Resource-Priority	(RFC 4412) AS-SIP req.
Retry-After	(RFC 3261)
RSeq	(RFC 3262) AS-SIP
Session-Expires (x)	(RFC 4028) AS-SIP
Subscription-State	(RFC 3265) AS-SIP
Supported (k)	(RFC 3261)
To (t)	(RFC 3261)
Unsupported	(RFC 3261)
Via (v)	(RFC 3261)
Warning	(RFC 3261)
WWW-Authenticate	(RFC 3261 generate and send only)

PRACK or Provisional Response Acknowledgement plays the same role as ACK except that it is for provisional responses only. PRACK is a standard SIP message and has its' own set of responses. It was surprisingly difficult to find true support for PRACK in most of the stacks we evaluated. IPSEC was not supported in any of the stacks we surveyed, commercial or open source.

4.2.3 AS-SIP Multilevel Precedence and Preemption (MLPP)

AS-SIP end instruments tag SIP communications with a precedence level or priority. This is accomplished by populating the "Resource-Priority" header. Resource-Priority consists of a namespace and a priority. The namespace is divided into 2 subfields separated by a "-". The value left of the "-" represents the "network domain" and the value to the right of the "-" is the "precedence domain". The "precedence domain" is currently set to "000000" for the time being. The namespace is then prefixed with ".priority" where priority is one of the following values:

0 Routine (lowest priority)
2 Priority
4 Immediate
6 Flash
8 Flash-Override (highest priority)

Example:

```
INVITE sip:user5001@192.168.0.138:54912 SIP/2.0
Via: SIP/2.0/UDP 192.168.0.11;branch=z9hG4bKe4580000056b90f5;rport
From: "ISDN Phone 2b" <sip:1015@192.168.0.11;user=phone>;tag=8a3b000000220d55b2c6
To: "SIP 5001" <sip:user5001@192.168.0.11;user=phone>
Contact: <sip:1015@192.168.0.11;CCA-ID=REDCOM;user=phone>
Call-ID: 7c405a43f9989320@192.168.0.11
CSeq: 159 INVITE
Max-Forwards: 70
Session-Expires: 3600;refresher=uac
```




Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Min-SE: 300

Allow:

REGISTER,INVITE,ACK,CANCEL,BYE,OPTIONS,INFO,REFER,SUBSCRIBE,NOTIFY,PR
ACK,
UPDATE

Allow-Events: dialog,message-summary

Supported: replaces,timer,100rel,resource-priority

Resource-Priority: dsn-000000.6

Content-Type: application/sdp

Content-Length: 347

User-Agent: REDCOM SLICE 2100 4.0a R3P3 18-Jun-2010

The example INVITE message above has a Resource-Priority header with a network domain of "dsn", a precedence domain of "000000" and a priority of "6" meaning this is a "FLASH" priority call. Our AS-SIP phone supported both the legacy dial prefix method of prioritizing a call and also took advantage of the software GUI to allow the user to visually select a call priority at the touch of a button. To manually set the priority of a call the dialed number is prefixed with one of the following 2 digits:

90 (FLASH OVERRIDE)

91 (FLASH)

92 (IMMEDIATE)

93 (PRIORITY)

94 (ROUTINE)

For our AS-SIP phone, these digits have no purpose and were simply stripped out and used to set the outbound Resource-Priority in software to whatever the user indicated on the fly. In the absence of these prefix digits, the priority is set to whatever was selected via the GUI on the phone. By default, the phone comes up with a priority of ROUTINE. It should be noted that with AS-SIP, much of the burden of supporting priority is shifted from the switch to the end instrument.

AS-SIP end instruments are required to support 2 lines. Preemption of a call does not occur unless both lines are occupied and a 3rd call arrives with a higher priority than either of the 2 lines in use and the arriving call has the same precedence domain as either of the calls on the lines in use. Preempted calls must be sent a Reason header with a cause code of 1 when the call is hung up. RFC 4411 defines 4 cause codes as shown below. Cause code 5 is a UCR requirement only and effects server side SIP appliances only.

Reason: preemption ;cause=1 ;text="UA Preemption"

Reason: preemption ;cause=2 ;text="Reserved Resources Preempted"

Reason: preemption ;cause=3 ;text="Generic Preemption"

Reason: preemption ;cause=4 ;text="Non-IP Preemption"

Reason: preemption; cause=5; text="Network Preemption"



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Example BYE with cause code = 1 to indicate to the receiving end instrument that the call was preempted:

```
BYE sip:user5001@192.168.0.138:54912 SIP/2.0
Via: SIP/2.0/UDP 192.168.0.11;branch=z9hG4bKe4580000056b90f5;rport
From: "ISDN Phone 2b" <sip:1015@192.168.0.11;user=phone>;tag=8a3b000000220d55b2c6
To: "SIP 5001" <sip:user5001@192.168.0.11;user=phone>
Contact: <sip:1015@192.168.0.11;CCA-ID=REDCOM;user=phone>
Call-ID: 7c405a43f9989320@192.168.0.11
CSeq: 159 INVITE
Max-Forwards: 70
Session-Expires: 3600;refresher=uac
Min-SE: 300
Allow:
REGISTER,INVITE,ACK,CANCEL,BYE,OPTIONS,INFO,REFER,SUBSCRIBE,NOTIFY,PR
ACK,
UPDATE
Reason: preemption ;cause=1 ;text="UA Preemption"
Allow-Events: dialog,message-summary
Supported: replaces,timer,100rel,resource-priority
Resource-Priority: dsn-000000.6
Content-Type: application/sdp
Content-Length: 347
User-Agent: REDCOM SLICE 2100 4.0a R3P3 18-Jun-2010
```

The Reason header with cause code=1 was used to trigger preemption events in our end instrument.

4.2.4 AS-SIP Precedence Rules

When a higher precedence call arrives with a different priority domain, no preemption occurs at the end instrument. The serving LSC is required to handle this scenario. The LSC is also responsible for handling preemption on single line end instruments and for handling a 480 response from the EI when the EI has no lines available and the caller is of lower precedence. Precedence occurs regardless of media type as well, meaning audio and video for example are treated equally.

When a call arrives of higher precedence than either of the 2 existing calls, the lower precedence call is preempted. The preempted call will display a preemption message as well as receive a preemption tone until the onhook is received. The called party also receives a preemption tone for a minimum of 3 seconds. The called party must also receive a preemption ring tone after going onhook on the preempted call. Upon going back off hook again, the called party must be connected to the priority call.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

When a higher precedence call arrives on a multi-line phone that still has one appearance available, then a precedence ring tone is provided however call preemption is entirely manual at this point and up to the called party to decide what to do with the call.

When the preempted call is a held call the preemption tone must still be played to the held call. The preemption ringtone is played immediately after the held call is dropped.

Preemption still applies to call forwarding in some instances. The UCR defines three call forward settings:

1. Call Forward Unconditional
2. Call Forward No Answer
3. Call Forward Busy.

In all instances, the precedence level of an incoming call must be preserved.

No preemption occurs when the phone is forwarded "unconditionally".

Preemption and precedence levels are still in force for call forward "busy".

Both the legacy method of setting call forward with manual prefix digits and the graphical user interface (GUI) to the SIP phone were supported. To manually set call forward one of the following dial prefixes were supported:

- | | |
|--------------------------------|-------------------------|
| 1. Call Forward Unconditional: | *72 (*73 to deactivate) |
| 2. Call Forward No Answer: | *92 (*93 to deactivate) |
| 3. Call Forward Busy: | *90 (*91 to deactivate) |

As with the preemption prefix digits, these digits had no meaning and were simply stripped out and used to set the call forward state of the SIP phone only. Users could also take advantage of the AS-SIP GUI and simply press a soft key to set or unset

The tangible differences between AS-SIP and SIP, those differences seen by the end user, could best be summed up as:

- Precedence level marking of each call
- Permission to use that marking
- Preferential treatment of calls
- The diversion (forwarding) of calls on no answer, busy and unconditional
- Preemption notifications
- The security and protection of both signaling and routing information

4.3 Open Source

A number of Open Source SIP stacks were considered as a starting point for our AS-SIP end instrument. At a minimum, we wanted stacks that already had support for IPV6, PRACK and secure mode. Open Source is a philosophy where producers make source and materials readily available to the end users. It promotes an environment where ideas are shared and users are



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

encouraged to build upon or improve the original source and materials. There are now millions of software packages and algorithms available to be downloaded free of charge via the internet. Only a few open source solutions had support for the minimum features we required for our AS-SIP endpoint application.

All of the open source SIP stacks surveyed and evaluated for the purpose of creating an AS-SIP end instrument, as defined by the DoD Unified Capabilities Requirements, were licensed under the GPL (GENERAL PUBLIC LICENSE) or some form of it (e.g. "Lesser GPL" or "GPL lite"). Most "open source" and/or "freeware" projects are covered by some form of the GNU GENERAL PUBLIC LICENSE (see Appendix A for a complete listing of the GPL agreement). Careful consideration should be given to the licensing agreement associated with an open source package before using or distributing copies of the software. Open source is generally considered to be "copyleft" protected. "Copyleft" was coined to deliberately contradict everything the word "copyright" stands for and implies. This is neither a warning nor endorsement of either software release practice. All parties involved should be aware of the unusual licensing agreement requirements of the GPL. Generally speaking, the GPL states that you must make the open source code, build procedures, and any other changes made available to whomever you distribute the code. You must document all changes and not distribute the code with objects that cannot be recreated (e.g. pre-compiled libraries). Users should also beware of the differences and consequences of software covered by the GPL vs. the LGPL. Generally speaking, Lesser GPL permits use of proprietary programs and libraries with the open source (see Appendix A for a complete listing of the Lesser GPL agreement).

4.3.1 Open Source SIP Stack Evaluations

4.3.1.1 oSIP

oSIP or *Open Session Initiation Protocol* is an implementation of SIP as described in RFC 3261. It is little more than an API for parsing SIP and SDP messages and is generally used as a starting point for the creation of both SIP based end instruments and or proxy/servers. The API is written entirely in "C", has a relatively small foot print and uses no other libraries outside of the standard "C" library. Its small size and portability makes OSIP equally well suited for both embedded and host based platforms. oSIP was created in September of 2000 and was compliant to the original SIP specification RFC 2543. It is a mature product and widely used. Our version of oSIP is based on RFC 3261, the latest SIP specification and also includes the newer API oSIP2 which was an attempt to make the API easier to use. The authors of oSIP warn of the difficulty in attempting to use oSIP. We found that the complexity involved with using oSIP was not so much the source itself, but the incompleteness of it in terms of creating a useful SIP application such as our endpoint. oSIP itself is only a small building block and requires numerous other open source packages in order to turn it into a useful SIP application client and or server. If we were building a car, it would be like starting with the engine only.

OSIP was designed to be platform independent which adds another significant level of complexity to what is already a very complex API. oSIP is an extremely granular package in nature and design. A simple voice SIP call for example, would require at least four other related



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

open source API's and a significant amount of code to implement the call. Whereas another open source SIP stack may include a single API function like "MakeCall()" to achieve the same goal of making a voice SIP call. There are considerable trade offs involved with any OS(open source) SIP package in terms of our goal of creating an "Assured Services" SIP end instrument. For example, while we may have been able to make a SIP voice call the day we downloaded and built the SipXTapi OS project, it did not lend itself very well to the modifications necessary to make it "Assured Services" compliant, especially in the area of additional RFC support. If we wanted to add support for the SIP header: "Resource-Priority" to the OSIP package it is done quite simply with a single API call. Adding support for that same SIP header to SipXTapi required a near reverse engineering effort of the entire package from the stack all the way up through their call manager to come up with a plan for implementing this simple feature. Numerous source files and hundreds of existing functions needed to be changed to add support for RFC4412(Resource-Priority) for example.

As previously stated, a number of related open source API's in addition to oSIP are required to make oSIP useful in terms of creating our AS-SIP end instrument.

The following opens source packages were downloaded, configured and compiled to build our AS-SIP endpoint:

- a. oRTP - RTP or Real Time Protocol, is a network protocol that defines a standard packet format for the delivery of audio and or video over the internet. oRTP is an implementation of RFC3550 (RTP) and also includes packet scheduling, IO multiplexing to support hundreds of sessions, an adaptive jitter algorithm and support for multiple profiles such as such as RFC3551 (Audio/Video).
- b. oSRTP - Secure RTP adds a layer of encryption, message authentication, integrity and replay protection to standard RTP and is also required for assured services SIP implementations.
- c. mediaStreamer2 - The mediaStreamer2 package provides an API for the handling of the RTP data streams. It provides audio/video codecs, filters, echo cancelation, dtmf generator, audio and video player and mixer.
- d. Open SSL – Open SSL implements the Secure Sockets Layer (SSL) as well as the Transport Layer Security (TLS), both of which are assured services requirements for our SIP end instrument device.
- e. eXosip2 - eXosip2 is a wrapper for oSIP and the other libraries and provides a much higher level set of API functions for creating SIP end instruments while still retaining the ease at which you can add support for additional RFC's. eXosip2 also includes much more support for additional RFC's. While not a complete "call manager" it does include high level functions for registering and telephony functions. Exosip2 shields the user from having to know the API of the other libraries and is for the most part the only API your application interfaces with. A simple example below illustrates the steps necessary to initialize and use the Exosip2 API to register:

```
//initialize SIP  
if (eXosip_init ())
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
{
    //error, exit
}

//set the port to listen on
if (eXosip_listen_addr (IPPROTO_UDP, NULL, [port number], AF_INET, 0))
{
    //error, exit
}

//set the user agent header string
eXosip_set_user_agent ("MY UA NAME");

//set the authentication info
if (eXosip_add_authentication_info(
username, username, password, NULL, NULL))
{
    //error, exit
}

//set and build the registration information
oSIP_message_t *reg = NULL;

int regid, expireTime;

if ((regid = eXosip_register_build_initial_register (
fromuser,
proxy,
contact,
expireTime, &reg)) < 1)
{
    //error, exit
}

//send the registration request
If (eXosip_register_send_register (regid, reg)
{
    //error, exit
}

//note that a threaded function must be created for continuous //re-registration, this can be done
via oSIP directly using:
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
oSIP_thread_create (20000, my_register_proc, &my_regparam);
```

```
//wait for SIP events
for (;;) //forever
{
    eXosip_event_t *event;

    if (!(event = eXosip_event_wait (0, 1)))

}

```

A brief description of the Exosip2 API's as follows

Exosip2 Configuration API

```
int eXosip\_init (void)
void eXosip\_quit (void)
int eXosip\_execute (void)
int eXosip\_set\_option (eXosip_option opt, const void *value)
int eXosip\_lock (void)
int eXosip\_unlock (void)
int eXosip\_get\_srv\_record (struct oSIP_srv_record *record, char *domain, char *protocol)
int eXosip\_listen\_addr (int transport, const char *addr, int port, int family, int secure)
int eXosip\_set\_socket (int transport, int socket, int port)
void eXosip\_set\_user\_agent (const char *user_agent)
const char * eXosip\_get\_version (void)
void eXosip\_enable\_ipv6 (int ipv6_enable)
void eXosip\_masquerade\_contact (const char *public_address, int port)
int eXosip\_find\_free\_port (int free_port, int transport)

```

Exosip2 Network API

```
int eXosip\_transport\_set (oSIP_message_t *msg, const char *transport)
int eXosip\_guess\_localip (int family, char *address, int size)

```

Exosip2 Event API

```
void eXosip\_event\_free (eXosip\_event\_t *je)
eXosip\_event\_t * eXosip\_event\_wait (int tv_s, int tv_ms)
eXosip\_event\_t * eXosip\_event\_get (void)
int eXosip\_event\_geteventsocket (void)

```

Events in Xosip2 are enumerated, the following events are defined:

EXOSIP_REGISTRATION_NEW announce new registration.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

EXOSIP_REGISTRATION_SUCCESS user is successfully registred.
EXOSIP_REGISTRATION_FAILURE user is not registred.
EXOSIP_REGISTRATION_REFRESHED registration has been refreshed.
EXOSIP_REGISTRATION_TERMINATED UA is not registred any more.
EXOSIP_CALL_INVITE announce a new call
EXOSIP_CALL_REINVITE announce a new INVITE within call
EXOSIP_CALL_NOANSWER announce no answer within the timeout
EXOSIP_CALL_PROCEEDING announce processing by a remote app
EXOSIP_CALL_RINGING announce ringback
EXOSIP_CALL_ANSWERED announce start of call
EXOSIP_CALL_REDIRECTED announce a redirection
EXOSIP_CALL_REQUESTFAILURE announce a request failure
EXOSIP_CALL_SERVERFAILURE announce a server failure
EXOSIP_CALL_GLOBALFAILURE announce a global failure
EXOSIP_CALL_ACK ACK received for 200ok to INVITE
EXOSIP_CALL_CANCELLED announce that call has been cancelled
EXOSIP_CALL_TIMEOUT announce that call has failed
EXOSIP_CALL_MESSAGE_NEW announce new incoming request.
EXOSIP_CALL_MESSAGE_PROCEEDING announce a 1xx for request.
EXOSIP_CALL_MESSAGE_ANSWERED announce a 200ok
EXOSIP_CALL_MESSAGE_REDIRECTED announce a failure.
EXOSIP_CALL_MESSAGE_REQUESTFAILURE announce a failure.
EXOSIP_CALL_MESSAGE_SERVERFAILURE announce a failure.
EXOSIP_CALL_MESSAGE_GLOBALFAILURE announce a failure.
EXOSIP_CALL_CLOSED a BYE was received for this call
EXOSIP_CALL_RELEASED call context is cleared.
EXOSIP_MESSAGE_NEW announce new incoming request.
EXOSIP_MESSAGE_PROCEEDING announce a 1xx for request.
EXOSIP_MESSAGE_ANSWERED announce a 200ok
EXOSIP_MESSAGE_REDIRECTED announce a failure.
EXOSIP_MESSAGE_REQUESTFAILURE announce a failure.
EXOSIP_MESSAGE_SERVERFAILURE announce a failure.
EXOSIP_MESSAGE_GLOBALFAILURE announce a failure.
EXOSIP_SUBSCRIPTION_UPDATE announce incoming SUBSCRIBE.
EXOSIP_SUBSCRIPTION_CLOSED announce end of subscription.
EXOSIP_SUBSCRIPTION_NOANSWER announce no answer
EXOSIP_SUBSCRIPTION_PROCEEDING announce a 1xx
EXOSIP_SUBSCRIPTION_ANSWERED announce a 200ok
EXOSIP_SUBSCRIPTION_REDIRECTED announce a redirection
EXOSIP_SUBSCRIPTION_REQUESTFAILURE announce a request failure
EXOSIP_SUBSCRIPTION_SERVERFAILURE announce a server failure
EXOSIP_SUBSCRIPTION_GLOBALFAILURE announce a global failure
EXOSIP_SUBSCRIPTION_NOTIFY announce new NOTIFY request
EXOSIP_SUBSCRIPTION_RELEASED call context is cleared.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

EXOSIP_IN_SUBSCRIPTION_NEW announce new incoming SUBSCRIBE.
EXOSIP_IN_SUBSCRIPTION_RELEASED announce end of subscription.
EXOSIP_NOTIFICATION_NOANSWER announce no answer
EXOSIP_NOTIFICATION_PROCEEDING announce a 1xx
EXOSIP_NOTIFICATION_ANSWERED announce a 200ok
EXOSIP_NOTIFICATION_REDIRECTED announce a redirection
EXOSIP_NOTIFICATION_REQUESTFAILURE announce a request failure
EXOSIP_NOTIFICATION_SERVERFAILURE announce a server failure
EXOSIP_NOTIFICATION_GLOBALFAILURE announce a global failure
EXOSIP_EVENT_COUNT MAX number of events

Exosip2 Invite/Call Management API

int [eXosip_call_set_reference](#) (int id, void *reference)
int [eXosip_call_build_initial_invite](#) (oSIP_message_t **invite, const char *to, const char *from, const char *route, const char *subject)
int [eXosip_call_send_initial_invite](#) (oSIP_message_t *invite)
int [eXosip_call_build_request](#) (int did, const char *method, oSIP_message_t **request)
int [eXosip_call_build_ack](#) (int did, oSIP_message_t **ack)
int [eXosip_call_send_ack](#) (int did, oSIP_message_t *ack)
int [eXosip_call_build_refer](#) (int did, const char *refer_to, oSIP_message_t **request)
int [eXosip_call_build_info](#) (int did, oSIP_message_t **request)
int [eXosip_call_build_options](#) (int did, oSIP_message_t **request)
int [eXosip_call_build_update](#) (int did, oSIP_message_t **request)
int [eXosip_call_build_notify](#) (int did, int subscription_status, oSIP_message_t **request)
int [eXosip_call_send_request](#) (int did, oSIP_message_t *request)
int [eXosip_call_build_answer](#) (int tid, int status, oSIP_message_t **answer)
int [eXosip_call_send_answer](#) (int tid, int status, oSIP_message_t *answer)
int [eXosip_call_terminate](#) (int cid, int did)
int [eXosip_call_build_prack](#) (int tid, oSIP_message_t **prack)
int [eXosip_call_send_prack](#) (int tid, oSIP_message_t *prack)
int [eXosip_transfer_send_notify](#) (int did, int subscription_status, char *body)
int [eXosip_call_get_refer_to](#) (int did, char *refer_to, size_t refer_to_len)
int [eXosip_call_find_by_replaces](#) (char *replaces)
(REFER and blind transfers)
int [eXosip_refer_build_request](#) (oSIP_message_t **refer, const char *refer_to, const char *from, const char *to, const char *route)
int [eXosip_refer_send_request](#) (oSIP_message_t *refer)

All of the above libraries were installed, configured and compiled for the Windows 2008 platform and integrated with an existing C# GUI phone application from Phase 1. We did not take this application any further than registering with the REDCOM switch due to time constraints. Most of the required libraries can be built for a number of platforms including:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- a. GNU/Linux
- b. MacOSX (Darwin)
- c. OpenBsd 3.1/3
- d. Windows NT/95/2000 (VC++6.0 or cygwin)
- e. Solaris
- f. HP-Unix
- g. VxWorks
- h. Embedded Linux
- i. WinCE

Some embedded systems with Linux.

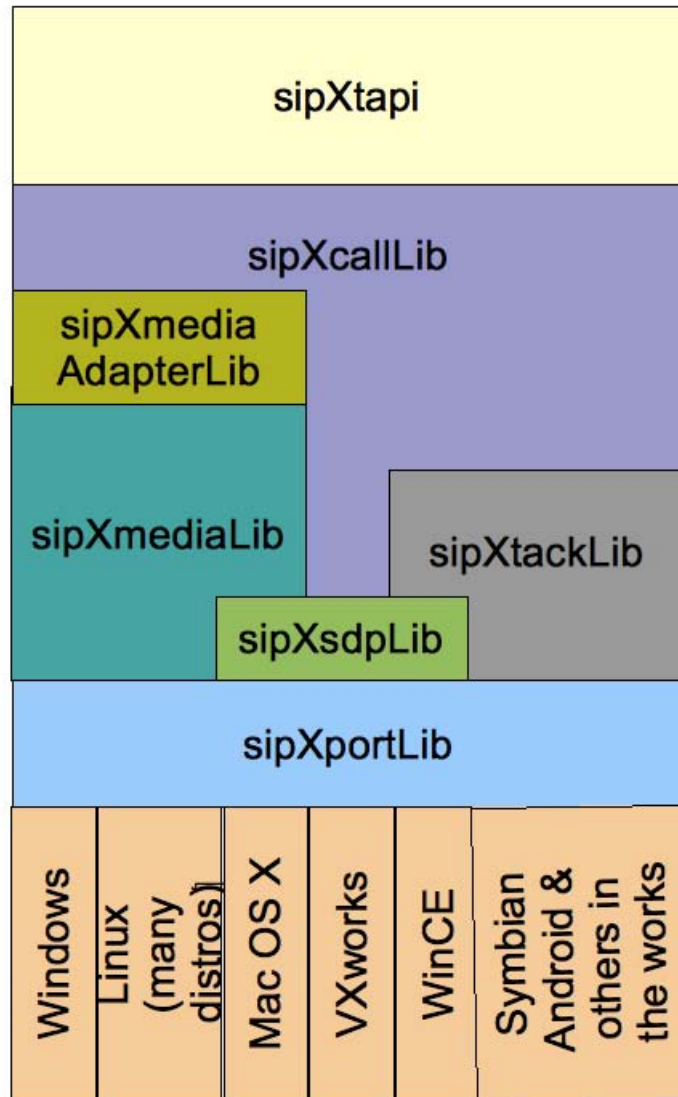
- a. WinCE (report to be possible)

Installing, compiling and linking these libraries together for Windows is like a puzzle, as there is no documentation that encompasses all of these open source libraries. Most of what can be included is controlled with `-D` compiler directives. The documentation and build process for Linux builds is considerably easier to use. Documentation in general for open source projects is extremely if not deliberately sparse as most open source providers expect to make their projects profitable on service and support contracts.

In conclusion, it should be noted that some of this code, most notably eXosip2, was not written or maintained within the United States. We were unable to find anyone who had used (enabled) the secure version(s) of these API's including the author of eXosip2, who resides in France. It was no small task to configure and compile all of these software packages into a single binary image. Documentation is minimal at best and support for open source packages is not free. "AntiSIP", the company who maintains most of these libraries charges a fee of \$3000 Euro per day for support.



sipX Suite of Libraries



SipXtapi, as illustrated above, is a complete open source SIP end instrument API from SIP Foundry that was an attractive choice because it included everything needed to create SIP end instrument applications in one single package. Both SIP signaling and media framework were all combined into a single project with support for many different codecs and platforms as shown in the **sipX Suite of Libraries** diagram above. SipXtapi is covered under the Lesser General Public License (LGPL) (see Appendix A for details).



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The original SipXtapi version we selected from SIP Foundry did not include support for PRACK, was somewhat buggy and had memory leaks. We did however find a re-factored version of the original sipXtapi API that included many fixes and enhancements: SipXtapi-3.3.0

SipXtapi 3.3.0 already had support for the following RFCs:

1. RFC 2782 - A DNS RR for specifying the location of services (DNS SRV) - <http://tools.ietf.org/html/rfc2782>
2. RFC 2833 - RTP Payload for DTMF Digits - <http://tools.ietf.org/html/rfc2833>
3. RFC 2976 - The SIP INFO Method - <http://tools.ietf.org/html/rfc2976>
4. RFC 3261 - SIP: Session Initiation Protocol - <http://tools.ietf.org/html/rfc3261>
5. RFC 3262 - Reliability of Provisional Responses in SIP - <http://tools.ietf.org/html/rfc3262>
6. RFC 3263 - Session Initiation Protocol (SIP): Locating SIP Servers - <http://tools.ietf.org/html/rfc3263>
7. RFC 3264 - An Offer/Answer Model with the Session Description Protocol (SDP) - <http://tools.ietf.org/html/rfc3264>
8. RFC 3265 - Session Initiation Protocol (SIP)- Specific Event Notification - <http://tools.ietf.org/html/rfc3265>
9. RFC 3311 - The Session Initiation Protocol (SIP) UPDATE Method - <http://tools.ietf.org/html/rfc3311>
10. RFC 3326 - The Reason Header Field for the Session Initiation Protocol (SIP) - <http://tools.ietf.org/html/rfc3326>
11. RFC 3420 - Internet Media Type message/sipfrag - <http://tools.ietf.org/html/rfc3420>
12. RFC 3428 - Session Initiation Protocol (SIP) Extension for Instant Messaging - <http://tools.ietf.org/html/rfc3428>
13. RFC 3489 - Simple Traversal of UDP through Network Address Translators (NATs) - <http://tools.ietf.org/html/rfc3489>
14. RFC 3515 - The Session Initiation Protocol (SIP) Refer Method - <http://tools.ietf.org/html/rfc3515>
15. RFC 3550 - RTP: A Transport Protocol for Real-Time Applications - <http://tools.ietf.org/html/rfc3550>
16. RFC 3551 - RTP Profile for Audio and Video Conferences with Minimal Control - <http://tools.ietf.org/html/rfc3551>
17. RFC 3581 - An extension to SIP for Symmetric Response Routing - <http://tools.ietf.org/html/rfc3581>
18. RFC 3891 - The Session Initiation Protocol (SIP) "Replaces" Header - <http://tools.ietf.org/html/rfc3891>
19. RFC 3903 - Session Initiation Protocol (SIP) Extension for Event State Publication - <http://tools.ietf.org/html/rfc3903>
20. RFC 3951 - Internet Low Bit Rate Codec (iLBC) - <http://tools.ietf.org/html/rfc3951>
21. RFC 3952 - Real-time Transport Protocol (RTP) Payload Format for internet Low Bit Rate Codec (iLBC) Speech - <http://tools.ietf.org/html/rfc3952>
22. RFC 4028 - Session Timers in the Session Initiation Protocol (SIP) -



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

<http://tools.ietf.org/html/rfc4028>

23. RFC 4488 - Suppression of Session Initiation Protocol (SIP) REFER Method Implicit Subscription - <http://tools.ietf.org/html/rfc4488>

24. RFC 4566 - SDP: Session Description Protocol - <http://tools.ietf.org/html/rfc4566>

25. RFC 4916 - Connected Identity in the Session Initiation Protocol (SIP) - <http://tools.ietf.org/html/rfc4916>

26. RFC 5057 - Multiple Dialog Usages in the Session Initiation Protocol - <http://tools.ietf.org/html/rfc5057>

27. Draft-ietf-mmusic-ice-04 - Interactive Connectivity Establishment (ICE)

a. draft-rosenberg-midcom-turn-04 - Traversal Using Relay NAT (TURN)

SipXtapi-3.3.0 included nearly all of the features we required to build an AS-SIP client including support for IPV6. The only missing piece was support for TLS. TLS was on the list of items to be supported in the future and we felt like we could probably contract with the author(s) to add secure mode at a later point. The following paragraphs describe the items (listed by library) that were added, corrected or enhanced in this branch.

4.3.1.2 sipXtapi

a. New API for getting all supported codecs. This was not available in 3.2.0 and had to be worked around (`sipxConfigGetNumAvailableAudioCodecs`, `sipxConfigGetAvailableAudioCodec`, `sipxConfigGetNumSelectedAudioCodecs`, `sipxConfigGetSelectedAudioCodec`)

b. Public and private conferences. Private conferences work like in 3.2.0. Public conferences are assigned a conference SIP URI. Dialing this SIP URI results in the call automatically created in a conference.

c. `sipxConferenceJoin`, `sipxConferenceSplit` don't require call hold

d. Conference events - see `SIPX_CONFERENCE_EVENT`, `SIPX_CONFERENCE_CAUSE` in `sipXtapiEvents`.

e. Call/media events will not be 100% compatible with 3.2.0. Any changes will be described to make upgrade to 3.3.0 easy.

f. Changes in call transfer events - transferee side (recipient of REFER request) gets DIALTONE event instead of NEWCALL, when creating new call, since that call is outbound.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- g. Possibility to accept/reject call transfer request (sipxCallAcceptTransfer, sipxCallRejectTransfer), no support for out of dialog REFER
- h. Safer sipxCallPlayBufferStart, buffer can be deleted immediately after function returns
- i. Selection of codecs by bandwidth in sipXtapi will not be supported. Only selection by name will be supported. User may display codecs along with bandwidth requirements, and then choose codecs by name.
- j. Separate functions sipxCallLimitCodecPreferences, sipxCallRenegotiateCodecPreferences
- k. sip proxy per line, using sipxLineSetOutboundProxy
- l. Better audio focus control when connecting call with SIPX_FOCUS_CONFIG
- m. contactId now works the way it should - when set SIPX_AUTOMATIC_CONTACT_ID then SipUserAgent will choose the best contact, otherwise contact selection will be respected (user knows better which contact is the best)
- n. Transport selection is functional - SIPX_CALL_OPTIONS has transportType which in cooperation with contactId selects transport. It is now possible to initiate a call using TCP transport which was not possible in 3.2.0. Also sipxConfigSubscribe, sipxCallSubscribe, sipxLineAdd, sipxPIMSendPagerMessage now have transport parameter.

4.3.1.3 sipXcallLib

- a. Conferencing without having to invoke hold/unhold on conference or call.
- b. Public and private conferences. Public conferences will be bound to line, and any accepted inbound calls will be added to conference
- c. Big refactoring in sipXcallLib, CpCall/CpPeerCall, Connection/SipConnection, CallManager/CpCallManager will be gone.
- d. Support for session timer (rfc4028), 100rel (rfc3262), norefersub (rfc4488), replaces (rfc3891), from-change (rfc4916), join (rfc3911)
- e. UPDATE, PRACK method support
- f. Early and late SDP negotiation (SDP in INVITE, or ACK), configurable
- g. Sending SDP in unreliable and reliable 18x responses - possible to send early audio
- h. configurable sending of reliable 18x responses, session timer and usage of UPDATE method for hold/unhold/codecs renegotiation
- i. refactored handling of inbound INFO



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- j. special message classes for messages instead of usage of `OsIntPtrMsg` or `CpMultiStringMessage`
- k. less circular dependencies, various OO design defects fixed
- l. old `sipXcallLib` is completely rewritten, since old code prevented development of new features

4.3.1.4 sipXsdpLib

- a. Minor refactoring of `sipXtackLib` and `sipXsdpLib` - SDP handling (`SdpCodecList`, `SdpCodecFactory`), usage of `SdpCodecList` in `sipXmediaAdapterLib` whenever possible, avoid passing of arrays
- b. Much easier to add new codecs. Codecs are still static, but the complexity of adding codecs has been reduced.

4.3.1.5 sipXtackLib

- a. Removed support for PING method, it never became a standard (<http://tools.ietf.org/html/draft-fwmiller-ping-03>)
- b. `SipMessage` class supports parsing and setting new fields
- c. Much smarter selection of sip contacts, it's no more needed to select them manually. This makes usage of `sipXtapi` in multi IP/multi-SIP proxy environment much easier

4.3.1.6 sipXmediaLib

- a. wide band audio support with 48Khz internal sampling rate. Old 8Khz narrow band is recommended for server applications due to lower CPU load.
- b. VAD (voice activity detection), DTX (discontinuous transmission) for all codecs, CNG (comfort noise generation)
- c. support for additional SPEEX modes (11,000 bps and 18,200 bps), almost all SPEEX wide band (16Khz), SPEEX ultra band modes (32Khz)
- d. iLBC 20ms and 30ms modes are supported for sending and receiving - 8Khz
 - G.711 u-law and a-law via built-in or Intel IPP 6.0 encoder/decoder
 - G.722 codec 64 Kbps via Span DSP library - 16Khz
 - G.722.1 codec (16, 24, 32 Kbps modes) via Intel IPP - 16Khz
 - G.723.1 codec (5.3, 6.3 Kbps, sending 5.3 Kbps) via Intel IPP - 8Khz
 - G.726 codec (16, 24, 32, 40 Kbps modes) via Span DSP library - 8Khz



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

G.728 sending 16 Kbps, receiving 12.8 Kbps and 16 Kbps via Intel IPP - 8Khz

G.729 annex A, B, D and E (with VAD or without) via Intel IPP - 8Khz

G.729.1 codec (8, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32 KBps, sending and receiving any rate) via Intel IPP - 16Khz

e. L16 audio - uncompressed 16bit mono audio in network byte order. All recommended sampling rates up to 48Khz.

f. GSM FR codec (13 Kbps) - via gsmlib or Intel IPP 6.0 encoder/decoder - 8Khz

h. AMR codec (4.75, 5.15, 5.9, 6.7, 7.4, 7.95, 10.2, 12.2 Kbps - sending 4.75/10.2 Kbps, receiving any rate), octet aligned and bandwidth efficient mode, via Intel IPP - 8Khz

i. AMR wideband codec (6.6, 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 Kbps - sending 12.65/23.85 Kbps, receiving any rate), octet aligned and bandwidth efficient mode, via Intel IPP - 16Khz

j. Improved old inbound DTMF detector, and new Span DSP DTMF detector (enabled by default in Windows). Span DSP DTMF detector only available when sipXtapi is compiled with only narrow band support. DTMF detection is mostly effective at 8Khz.

k. Upgrade to Intel IPP 6.0 (compatibility with Intel IPP 5.3 maintained)

l. Upgrade to SPEEX 1.2rc1 library

m. Usage of synchronous Portaudio API instead of asynchronous. Reduction of latency by ~160ms by this change.

n. Possibility to adjust audio driver latency

o. Linear complexity bridge (less CPU cost)

p. introduction of Span DSP 0.0.6pre3

q. media events MEDIA_REMOTE_SILENT, MEDIA_REMOTE_ACTIVE work

4.3.1.7 sipXportLib

a. Minor improvements in sipXportLib - OsSharedServerTask, OsRWMutex inherits from OsRWSyncBase and is write recursive. OsQueuedNotification superseding OsQueuedEvent.

b. Usage of OsTimerNotification for sending timer event messages. Custom notification subclass for each timer, avoiding any memory leaks.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

4.4 Building SipXtapi

Our application was built with MS Dev Studio 2008 on a Windows XP platform. Our first few attempts to build 3.3 did not go very well. The application had both 2005 and 2008 project files; however it would not build under 2005. The 3.3 was also still very much a developmental build and we were forced to contact the author to help address a number of problems we had with missing features. The author, who we knew by email handle only: "jaroslav11" was quite helpful throughout our trials with 3.3. We received a number of new builds of the package from him over the first few weeks. The source for 3.3 is organized as follows:

```
$INSTALL_DIR    \sipXtapi
                  \config
                  \contrib
                  \sipXcallLib
                  \sipXmediaAdapterLib
                  \sipXmediaLib
                  \sipXportLib
                  \sipXsdpLib
                  \sipXtackLib
                  *\GUI
```

Note that the "GUI" project is not part of the sipXtapi 3.3 package. The GUI was the same soft phone interface created for phase 1. It is a C#/.net soft phone application and consequently “managed” code. The sipXtapi3.3 source is straight C/C++ unmanaged code meant to be built for native Windows and several other platforms as well. A significant marshalling layer had to be created to integrate the managed GUI code with sipXtapi.

We added only 2 source files to the existing sipXtapi 3.3 distribution:

```
$INSTALL_DIR\sipXtapi\sipXcallLib\examples\PlaceCall\src\AsSipCli.c
$INSTALL_DIR\sipXtapi\sipXcallLib\examples\PlaceCall\include\AsSipCli.h
```

AsSipCli.c could be built as a standalone DOS command line application or as a library for our GUI application by defining the macro preprocessor "ASIP_CONSOLE_APP" or not:

```
#if defined (ASIP_CONSOLE_APP)
int main(int argc, char* argv[])
{
...
#else
void asSipXInitApi( int line,
                   char *proxy,
                   char *fromId,
                   char *userAuth,
                   char *passwdAuth,
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
char *realm,  
int sipPort,  
int rtpPort)  
  
{  
...  
}
```

This console application was especially handy for testing new functionality and for automated tests such as repetitive call scenarios.

The following command line options were available to the console version of our client:

```
-s SIP port (default = 5060)  
-r RTP port start (default = 9000)  
-u username (for authentication)  
-p password (for authentication)  
-m realm (for authentication)  
-f from identity  
-x proxy (outbound proxy)  
-st stun server  
-b ip address to bind to  
-rp use rport as part of via (disabled by default)  
-i call input device name  
-o call output device name  
-c codec name  
-t start app in console test mode  
-tx <uri> auto transfer call after answer  
-nd number of calls to make, console test mode only  
-lf log file name, stdout by default  
-h display this help\n");
```

Project files:

To build the console application:

```
$INSTALL_DIR\sipXtapi\sipXcallLib\examples\PlaceCall\PlaceCall-msvc9.vcproj
```

To build the GUI application:

```
$INSTALL_DIR\sipXtapi\GUI\GUI.csproj
```

Application Files:

```
$INSTALL_DIR\sipXtapi\sipXcallLib\examples\PlaceCall\src\AsSipCli.c  
$INSTALL_DIR\sipXtapi\sipXcallLib\examples\PlaceCall\include\AsSipCli.h
```

As previously mentioned, these two files were the only new source files we added to sipXtapi. AsSipCli.c was our application layer and interface between sipXtapi and our existing GUI softphone application.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Initializing sipXtapi was fairly straight forward and is best illustrated with code fragments from our console application as it does not require human intervention:

```
//  
//set the desired level of debugging (OPTIONAL)  
//  
sipxConfigSetLogLevel(LOG_LEVEL_DEBUG);  
  
//  
//designate the log file name and location (OPTIONAL)  
//  
sipxConfigSetLogFile("SIPXTAPI.log");  
  
//  
//Initialize sipXtapi (MANDATORY)  
//  
sipxInitialize( &gCFG.sipxHdl,  
               gCFG.sipPort,  
               gCFG.sipPort,  
               -1,  
               gCFG.rtpPort,  
               DEFAULT_CONNECTIONS,  
               DEFAULT_IDENTITY,  
               gCFG.bindAddr);  
  
//  
//enable PRACK, this is optional but required for AS-SIP  
//  
sipxConfigSet100relSetting( gCFG.sipxHdl,  
                           SIPX_100REL_REQUIRE_RELIABLE);  
  
//  
//this installs our own call back function handler for unsolicited sipXtapi events  
//  
sipxEventListenerAdd(gCFG.sipxHdl, EventCallBack, NULL);  
  
//configure rtp port  
sipxConfigEnableRport(gCFG.sipxHdl, true);  
  
//configure proxy server  
sipxConfigSetOutboundProxy(gCFG.sipxHdl, gCFG.proxy);  
if (gCFG.stunServer)
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
//optionally configure for STUN server
sipxConfigEnableStun(gCFG.sipxHdl, gCFG.stunServer,
                    DEFAULT_STUN_PORT, 28);

//optionally configure an alternative audio output device
sipxAudioSetOutputDevice(gCFG.sipxHdl, gCFG.outputDev)

//optionally configure an alternative audio input device
sipxAudioSetInputDevice(gCFG.sipxHdl, gCFG.inputDev

//enable echo cancel, noise reduction ect...
sipxAudioSetAECMode(gCFG.sipxHdl, SIPX_AEC_CANCEL_AUTO);
sipxAudioSetAGCMode(gCFG.sipxHdl, true);
sipxAudioSetNoiseReductionMode(gCFG.sipxHdl,
                               SIPX_NOISE_REDUCTION_DISABLED);

asStartSipClient();

appLog(1, "Registering Line 1...\n");
if (! asSipXRegister(gCFG.fromIdentity, gCFG.userName, gCFG.passWord,
gCFG.realm, &gCFG.lineHdl))
    appLog(1, "ERROR: unable to register\n");

// and finally, a threaded routine to wait for unsolicited “reactive” events from sipXtapi
//or user initiated “active” events from the phone.
//wait for application events

asWaitForEvents(LPVOID lpParam);
```

Our “wait” handler above was only necessary to keep the application up while waiting for phone activity. It actually only waited for 2 events, a shutdown from sipXtapi or a fatal error that would implicitly cause a shutdown of the program.

Our sipXtapi application was divided into 3 major components: reactive, active and a marshalling layer to integrate the unmanged C/C++ sipXtapi application with the managed C#/.net GUI phone application.

Reactive sipXtapi Control

The “reactive” portion of our application encompasses all of the unsolicited asynchronous events generated by the SipXtapi API during phone operation and the code we wrote to handle such events.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

SipxTapi events are divided into 13 enumerated categories:

```
EVENT_CATEGORY_CALLSTATE
EVENT_CATEGORY_LINESTATE
EVENT_CATEGORY_INFO_STATUS
EVENT_CATEGORY_INFO
EVENT_CATEGORY_SUB_STATUS
EVENT_CATEGORY_NOTIFY
EVENT_CATEGORY_CONFIG
EVENT_CATEGORY_SECURITY
EVENT_CATEGORY_MEDIA
EVENT_CATEGORY_KEEPAIVE
EVENT_CATEGORY_RTP_REDIRECT
EVENT_CATEGORY_CONFERENCE
EVENT_CATEGORY_PIM
```

Each category in turn had its own set of enumerated states and causes specific to the nature of the event category. For example, the “call state” category (EVENT_CATEGORY_CALLSTATE), the most extensive by far, has a rich set of call related event cause and state information as shown in the code fragments below.

It is within each of these event categories where the application developer can tailor the behavior of the SIP end instrument application.

To demonstrate how reactive sipXtapi events are parsed and handled we look at the event category “CALLSTATE” below. Note that each of the 13 event categories have the exact same look and feel as the call state handler detailed below.

In the first set of call state event information, we parse the “cause” of the particular call Event:

```
//
//CALL event handler
//
static void procCallState(SIPX_CALLSTATE_INFO *callState, char *evtName)
{
    char
        cEvent[32],
        cCause[32];
    //
    //get the cause code for the call state event
    //
    switch (callState->cause)
    {
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
//Unknown cause
case CALLSTATE_CAUSE_UNKNOWN:
    strcpy(cCause, "UNKNOWN");
    break;

//state changed due to normal operation
case CALLSTATE_CAUSE_NORMAL:
    strcpy(cCause, "NORMAL");
    break;

//call is being transferred to this UA
case CALLSTATE_CAUSE_TRANSFERRED:
    strcpy(cCause, "TRANSFERRED");
    break;

//this UA is transferring his call
case CALLSTATE_CAUSE_TRANSFER:
    strcpy(cCause, "TRANSFER");
    break;

//conference operation caused a state change
case CALLSTATE_CAUSE_CONFERERENCE:
    strcpy(cCause, "CONFERERENCE");
    break;

//The remote party is alerting and providing
//ringback audio (early media)
case: CALLSTATE_CAUSE_EARLY_MEDIA:
    strcpy(cCause, "EARLY_MEDIA");
    break;

//The callee rejected a request (e.g. hold)
case CALLSTATE_CAUSE_REQUEST_NOT_ACCEPTED
    strcpy(cCause, "REQ_REJECTED");
    break;

//Bad URL address or problem with DNS server?
case CALLSTATE_CAUSE_BAD_ADDRESS:
    strcpy(cCause, "BAD_URL");
    break;

case CALLSTATE_CAUSE_BUSY:
    //remote part BUSY
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
        strcpy(cCause, "BUSY");
        break;

//Not enough resources are available for operation
case CALLSTATE_CAUSE_RESOURCE_LIMIT:
        strcpy(cCause, "RESOURCE_LIMIT");
        break;

//A network error caused the desired operation to fail
case CALLSTATE_CAUSE_NETWORK:
        strcpy(cCause, "NETWORK_ERROR");
        break;

//The state changed due to a REDIRECTION of a call.
case CALLSTATE_CAUSE_REDIRECTED:
        strcpy(cCause, "REDIRECTED");
        break;

//No response was received from the remote party or network
case CALLSTATE_CAUSE_NO_RESPONSE:
        strcpy(cCause, "NO_RESPONSE");
        break;

//Unable to authenticate, bad or missing credentials
case CALLSTATE_CAUSE_AUTH:
        strcpy(cCause, "AUTHENTICATE_ERROR");
        break;

//A transfer(blind or warm) attempt has been initiated.
case CALLSTATE_CAUSE_TRANSFER_INITIATED:
        strcpy(cCause, "XFER_INITIATED");
        break;

//remote UA accepted the TRANSFER (REFER method)
case CALLSTATE_CAUSE_TRANSFER_ACCEPTED:
        strcpy(cCause, "XFER_ACCEPTED");
        break;

//still attempting the transfer
case CALLSTATE_CAUSE_TRANSFER_TRYING:
        strcpy(cCause, "XFER_ATTEMPT");
        break;
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
//The transfer target is ringing. BLIND transfers only
//consults or warm xfers go directly to success or fail
case CALLSTATE_CAUSE_TRANSFER_RINGING:
    strcpy(cCause, "XFER_RINGING");
    break;

//transfer was completed successfully.auto disconnect
case CALLSTATE_CAUSE_TRANSFER_SUCCESS:
    strcpy(cCause, "XFER_SUCCESS");
    break;

//transfer failed, APP must recover the HELD call!
case CALLSTATE_CAUSE_TRANSFER_FAILURE:
    strcpy(cCause, "XFER_FAIL");
    break;

//remote party does not support S/MIME
case CALLSTATE_CAUSE_REMOTE_SMIME_UNSUPPORTED:
    strcpy(cCause, "NOSUPPORT_SMIME_REMOTE");
    break;

//S/MIME operation failed. Should get SECURITY event
case CALLSTATE_CAUSE_SMIME_FAILURE:
    strcpy(cCause, "SMIME_ERROR");
    break;

//sipXtapi shutdown.
//depreciated
//case CALLSTATE_CAUSE_SHUTDOWN:

//unusable REFER was sent to this user-agent.
case CALLSTATE_CAUSE_BAD_REFERER:
    strcpy(cCause, "BAD_REFERER");
    break;

//This user-agent received a request or response,
//with no known matching invite. ????
case CALLSTATE_CAUSE_NO_KNOWN_INVITE:
    strcpy(cCause, "NO_INVITE");
    break;
```




Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
//BYE (hangup)message received but UA is idle
case CALLSTATE_CAUSE_BYE_DURING_IDLE:
    strcpy(cCause, "IDLE_BYE");
    break;

//response was received with an unknown status code
case CALLSTATE_CAUSE_UNKNOWN_STATUS_CODE:
    strcpy(cCause, "UNKNOWN_STATUS");
    break;

//received a redirect with NO contact or a RANDOM redirect.
case CALLSTATE_CAUSE_BAD_REDIRECT:
    strcpy(cCause, "BAD_REDIRECT");
    break;

//Attempt to Accept/Reject call not part of xfer.
case CALLSTATE_CAUSE_TRANSACTION_DOES_NOT_EXIST:
    strcpy(cCause, "NO_XFER_TRANSACTION");
    break;

//response to a cancel from the remote party
case CALLSTATE_CAUSE_CANCEL:
    strcpy(cCause, "REMOTE_CANCEL");
    break;

//An attempt to send INVITE with no codecs in SDP
//depreciated
//case CALLSTATE_CAUSE_NO_CODECS:

//unknown 4xx response
case CALLSTATE_CAUSE_CLIENT_ERROR:
    strcpy(cCause, "CLIENT_ERROR4xx");
    break;

//unknown 5xx response
case CALLSTATE_CAUSE_SERVER_ERROR:
    strcpy(cCause, "SERVER_ERROR5xx");
    break;

//unknown 6xx response
case CALLSTATE_CAUSE_GLOBAL_ERROR:
    strcpy(cCause, "GLOBAL_ERROR6xx");
    break;
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
//CAUSE is not applicable for some events
default:
    strcpy(cCause, "NA");
    break;
}
```

The next step in processing the call event is to parse the current “state” of the call. Note that all of the event categories are processed in much the same manner, with both cause and state information made available to the application developer as applicable.

```
//
//get the call state information for the call event
//
switch ( callState->event )
{

//unexpected error
case CALLSTATE_UNKNOWN:
    strcpy(cEvent, "UNKNOWN");
    break;

//new INBOUND call, automatically created by sipxTapi
case CALLSTATE_NEWCALL:
    strcpy(cEvent, "NEW CALL");
    break;

//new OUTBOUND call, app: simulate DIALTONE for the caller
case CALLSTATE_DIALTONE:
    strcpy(cEvent, "DIAL TONE");
    break;

//call setup sent
case CALLSTATE_REMOTE_OFFERING:
    strcpy(cEvent, "REMOTE OFFER");
    break;

//call setup accepted, end user RINGING, (3 minute limit?)
//If cause code = CALLSTATE_CAUSE_EARLY_MEDIA, the remote the party is
//sending early media meaning the Gateway(switch) is producing ringback or //audio
feedback
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
case CALLSTATE_REMOTE_ALERTING:
    strcpy(cEvent, "REMOTE ALERTING");
    break;

//BRIDGED state means call is active, but the local microphone/speaker are not
//engaged. If call is part of a conference, the party will be able to talk with other
//BRIDGED conference parties. Application developers can still play and record
//media.
case CALLSTATE_BRIDGED:
    strcpy(cEvent, "BRIDGED");
    break;

//call is both locally and remotely held. No network audio is flowing and the local
//microphone and speaker are not engaged.
case CALLSTATE_HELD:
    strcpy(cEvent, "HELD");
    break;

//remote party is on hold. Locally, the microphone and speaker are still
//engaged, however, no network audio is flowing.
case CALLSTATE_REMOTE_HELD:
    strcpy(cEvent, "REMOTE_HELD");
    break;

//a new call invitation has been extended this user agent. Application developers
//MUST invoke sipxCallAccept(), sipxCallReject() or sipxCallRedirect() in
//response. Not responding will result in an implicit call sipXcallReject().
case CALLSTATE_OFFERING
    strcpy(cEvent, "OFFERING");
    appLog(1, "->accepting the call offered\n");
    sipxCallAccept(callState->hCall);
    break;

//an inbound call has been accepted and the application layer should
//alert(RING!) the end user. The alerting state is limited to 3 minutes before it will
//be canceled
case CALLSTATE_ALERTING:
    strcpy(cEvent, "ALERTING");
    appLog(1, "->answering the call alerting\n");
    sipxCallAnswer(callState->hCall);
    break;

//call has been setup between local/remote party. Network audio should be
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
//flowing provided and the microphone and speakers should be engaged.
case CALLSTATE_CONNECTED:
    strcpy(cEvent, "CONNECTED");
    //INBOUND speak/play a file..
    break;

//call was disconnected or failed to connect. A call may move into the
//DISCONNECTED states from almost every other state. See DISCONNECTED
//minor events to understand the cause.
case CALLSTATE_DISCONNECTED:
    strcpy(cEvent, "DISCONNECTED");
    //INBOUND
    appLog(1,"->Destroying the disconnected call\n");
    asEndCall(&callState->hCall) ;
    break;

//last event apps will receive for any call, indicates all the underlying resources
//have been freed. Call handle no longer valid.
case CALLSTATE_DESTROYED:
    strcpy(cEvent, "DESTROYED");
    break;

//indicates a state change in the transfer attempt. See the
//CALLSTATE_TRANSFER_EVENT cause codes for details
case CALLSTATE_TRANSFER_EVENT:
    strcpy(cEvent, "XFER_EVENT");
    break;

//unexpected error
default:
    strcpy(cEvent, "UNEXPECTED");//ERROR
    break;
}

//if command line test mode...
if ((gCFG.genCallsCount) && (callState->event ==
                                                                    CALLSTATE_CONNECTED))
{
    appLog(1,"Sleeping 3 seconds before hold call\n");
    SLEEP(3000);
    asHoldCall( gCFG.callHdl );
}
else if ((gCFG.genCallsCount) && (callState->event ==
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```

CALLSTATE_HELD))

{
    appLog(1,"Sleeping 3 seconds before unhold call\n");
    SLEEP(3000);
    gCFG.genCallsCount =0;
    asUnHoldCall( gCFG.callHdl );
}

}

```

All of the event categories are processed nearly the same way as the call state category handler shown above. Each category has a rich set of events, causes and any applicable data specific to each event. It is within each of these 13 event categories that you build the receive side (reactive) of your SIP phone application. The call back handler you provide to SipXtapi is already threaded for you. The following are the handlers we developed for each SipXtapi event category:

```

static void procCallState(SIPX_CALLSTATE_INFO *callState, char *evtName);
static void procLineState(SIPX_LINESTATE_INFO *lineState, char *evtName);
static void procInfoStat(SIPX_INFOSTATUS_INFO *infoStat, char *evtName);
static void procInfoInfo(SIPX_INFO_INFO *infoData, char *evtName);
static void procSubStatus(SIPX_SUBSTATUS_INFO *subStatData, char *evtName);
static void procNotify(SIPX_NOTIFY_INFO *notifyData, char *evtName);
static void procConfig(SIPX_CONFIG_INFO *configData, char *evtName);
static void procSecurity(SIPX_SECURITY_INFO *pSecData, char *evtName);
static void procMedia(SIPX_MEDIA_INFO *pMediaData, char *evtName);
static void procKeepAlive(SIPX_KEEPA_LIVE_INFO *pKeepAlvData, char *evtName);
static void procPagerMsg( SIPX_PIM_INFO *pPimData, char *evtName);
static void procRtpRedirect(SIPX_RTP_REDIRECT_INFO *redirData, char *evtName);
static void procConfData(SIPX_CONFERENCE_INFO *confData, char *evtName);

```

4.4.1 Active SipXtapi Control

Active call control is accomplished using the high level call function API provided by SipXtapi combined with our own end instrument application requirements and features. Our active control begins with a set of enumerated call functions that are used by the GUI to indicate what phone function the phone would like to make. The following enumerations are the functions supported:

```

typedef enum ASSIP_COMMANDS
{
    COMMAND_RESPONSE           = 0,
    COMMAND_MAKE_CALL          = 1,
    COMMAND_ANSWER_CALL        = 2,
    COMMAND_HANGUP              = 3,

```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

COMMAND_HOLD	= 4,
COMMAND_UN_HOLD	= 5,
COMMAND_REFER	= 6,
COMMAND_TRANSFER	= 6,
COMMAND_FORWARD	= 7,
COMMAND_REREGISTER	= 90,
COMMAND_RESTART	= 91,
COMMAND_UNREGISTER	= 92,
COMMAND_SERVICE_FAILURE	= 93,
COMMAND_LOCAL_STATUS	= 94,
COMMAND_UNRECOGNIZED	= 95

```
}ASSIP_COMMANDS;
```

The GUI application indicates what it wants to by setting the command to one of the values above. These enumerations in turn cause the invocation of one of the following active control support functions:

```
//initializes sipXtapi
```

```
oid asSipXInitApi(    int line,  
                     char *proxy,  
                     char *fromId,  
                     char *userAuth,  
                     char *passwdAuth,  
                     char *realm,  
                     int sipPort,  
                     int rtpPort);
```

```
static bool asStopSipXClient();
```

```
static bool asSipXRegister(char *fromId, char *user, char *passwd,  
                           char *realm, SIPX_LINE *lineHdl);
```

```
void asSipXLineInitApi(    int line, char *proxy, char *fromId, char *userAuth,  
                           char *passwdAuth, char *realm);
```

```
static void asEndCall(SIPX_CALL* hCall);
```

```
static bool asMakeCall(char* url, char* from, char* user, char* pass, char *realm,  
                       char* precedence_domain,  
                       char* resource_priority, SIPX_LINE lineHdl,  
                       SIPX_CALL *callHdl);
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
static bool asPlayTones(char* toneStr, SIPX_CALL callHdl);

static bool asPlayFile(char* szFile);

static bool asTransferCall(const SIPX_CALL hCall, char* url,
                           char* precedence_domain, char* resource_priority);

static SIPX_RESULT asHoldCall(SIPX_CALL* hCall);

static SIPX_RESULT asUnHoldCall(SIPX_CALL* hCall);
```

The following code fragment is our handler for commands from the GUI application, this is largely what we call the “active” SipXtapi control. The “method” argument below indicates one of the previously mentioned ASSIP commands. The unusual function declaration is how functions are marshalled (see paragraph 4.4.2) between managed and unmanaged code:

```
extern "C" __declspec(dllexport) void __stdcall sipX_dll_voip_phone_send_command(
    int connid,
    int method,
    int response,
    char *arg1, char *arg2, char *arg3, char *arg4)
{
    int
        callState;
    SIPX_CALL
        callHandle;
    SIPX_LINE
        lineHandle;
    SIPX_RESULT
        result = SIPX_RESULT_SUCCESS;

    appLog(1, "RECV GUI EVT: <command %d> %s line_id:%d\n", method,
        guiCmd2Str((ASSIP_COMMANDS)method), connid);

    gCFG.appLine = connid;
    gCFG.precedence_domain = arg3;

    //need this string around OUTSIDE the scope of this funtion
    strcpy(gCFG.resource_priority, arg4);
    sipxSetResourcePri( gCFG.resource_priority );
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
callState = get_CallStateCurrent(connid);
callHandle = get_CallHandle(connid);
lineHandle = get_LineHandle(connid);

switch(method)
{
    case COMMAND_RESPONSE: // = 0, /* pseudo-method for responses */
        switch(response)
        {
            case 200: //ok
                appLog(1, "sendCommand->response->ok (accepting the
call offered)\n");

                createCallInfo(gCFG.callHdl, lineHandle, connid);
                sipxCallAnswer(gCFG.callHdl);
                break;
            case 486: //BUSY HERE
                appLog(1, "sendCommand->response->Busy Here\n");
                break;
            default:
                appLog(1, "sendCommand->response->Unhandled:
<%d>\n", response);

                break;
        } //Response Switch

        break;
    case COMMAND_MAKE_CALL: // = 1,
        gCFG.url = arg1;
        if (asMakeCall(gCFG.url, gCFG.fromIdentity, gCFG.userName,
gCFG.passWord, gCFG.realm,
gCFG.precedence_domain,
gCFG.resource_priority,
gCFG.lineHdl, &gCFG.callHdl))
        {
            createCallInfo(gCFG.callHdl, lineHandle, connid);
            appLog(1, "asMakeCall() OK\n");
        }
        else
        {
            appLog(1, "asMakeCall() FAILED\n");
        }
        break;
    case COMMAND_ANSWER_CALL: // = 2,
        appLog(1, "sendCommand->AnswerCall (accepting the call offered)\n");
        createCallInfo(gCFG.callHdl, lineHandle, connid);
```




Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
        sipxCallAnswer(gCFG.callHdl);  
        break;  
//code fragment end
```

4.4.2 Marshalling Layer

The marshalling layer of the software completed the interface between the C#/.NET GUI code and our SipXtapi application. This interface serviced both active and reactive SipXtapi telephony operations. Again, this layer was necessary because sipXtapi is what Microsoft refers to as ‘unamanged’ source code compiled to run on native Windows platforms. Ideally, applications should be either all managed or all unmanaged but where the .NET (dot-net) environment is relatively new Microsoft recognized the need to provide a means for integrating with traditional Windows applications. The GUI code was written in C# for the .NET platform. It essentially runs inside a virtual machine much the same way JAVA runs inside the JRE (JAVA Runtime Environment) with the goal of making the code portable. Our interface to the GUI was handled with four simple marshaled functions:

To initialize the SipXtapi stack from the GUI:

```
extern "C" __declspec(dllexport) void __stdcall sipX_dll_voip_system_init(  
    char *appName,  
    char *szBindAddr,  
    char *szStunServer,  
    int iSipPort,  
    int iRtpPort  
)
```

To initiaize a line from the GUI:

```
extern "C" __declspec(dllexport) void __stdcall sipX_dll_voip_line_init(  
    char *szUsername,  
    char *szAuthUsername,  
    char *szPassword,  
    char *szRealm,  
    char *szFromIdentity,  
    char *szProxy,  
    char *szCodecName )
```

To receive events from SipXtapi. Events in sipXtapi are queued into a simple FIFO and made available to the GUI:

```
extern "C" __declspec(dllexport) void __stdcall sipX_dll_PopEvent(int *asCategory,  
    int *asEvent,  
    int *asCause,  
    int *asLine,
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
int *asCall,  
char *anyStr,  
char *szArg1,  
char *szArg2,  
char *szArg3,  
char *szArg4)
```

To receive phone functions from the GUI:

```
extern "C" __declspec(dllexport) void __stdcall sipX_dll_voip_phone_send_command(  
int connid,  
int method,  
int response,  
char *arg1,  
char *arg2,  
char *arg3,  
char *arg4)
```

4.5 Conclusion

The work done with the open sources was an initial review to see how the three methods of implementing AS-SIP could be managed:

- Can the AS-SIP be added to the stack directly,
- Can AS-SIP be added as a “bolt-on” appendage,
- Or some combination of the two.

There are a couple of variables that influence the decision of how to maintain the project and how to return it (if applicable) to the community. As we saw with the SipXtapi several communities don't want the code back if structural changes have been made to the base line code. Many of the established open source projects have been released from other projects that do not want the baseline to change since it will cause them to restructure the other projects. In this case you are adding major functionality that may be good for your needs but has no value for the open community. As in the case of SipXtapi, one can create a branch of the open source project and build one's own community.

Another issue with the open community is the ability to make sure there is no malaise or *Trojan* code. A major issue is cleaning the project and then maintaining a tight review on its introduction. The UCR is on the Web, but the document is meant for military use only. So the releasing of the open source project to the community (the Internet) does not really support that intent. If the open source package was released into the Military-only intranet, then it must be released under its own license; it must also meet the requirements for the license that it was



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

released under. This would allow the military and its sub contractors to work from a single AS-SIP stack that could be tested once and maintained. This could reduce the amount of testing that an end device would have to undergo for its JITC testing, if the AS-SIP stack is known to be correctly implemented.

The bolt-on solution could leave the open source in the community that created it, but would require continuous validation of the cleanliness of the code for no malaise or Trojan code. The code would need to have functions, call backs and hooks added to the code that allowed the bolt-on code to receive the AS-SIP messages, and to allow for changing the headers and parameters before they are sent out. The added layers of code and indirection would result in performance reductions, but for this type of application that should not be a major issue. The place that may be affected is in large applications that have many individual stacks running at the same time to support multiple calls simultaneously. This would be like an automatic audio-attendant. The benefits for this type of implementation are outweighed by the limitations of having to react after the receiving of the message.

The implementation for an open source community stack would start from oSIP and integrate all the individual components into the solution. In the beginning this method would take a lot longer, but the stack could be modified with the functionality built directly into the state machine allowing for more flexibility as well as improved performance for the final stack. This solution would only be released to the military community and its sub contractors. This branch would be separated from the main branch and would never enjoy the benefits of the open community's updates. Because of the structural nature of the changes, the ability to merge open community changes into this branch would be prone to major issues since the two code bases would quickly diverge; making common areas to merge almost impossible to find, or at least unmanageable.

Our recommendation, if the direction is from an open source project, would be to start with an project like oSIP and its supporting open source projects and build the AS-SIP call manager with AS-SIP coded directly into the stack. Subsequently, it could be moved to the intranet so other military groups and sub contractors could contribute to the project. With the right supporters and JITC support, a call manager could be developed that would allow for a single project that is pre-certified by JITC.



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

This page intentionally left blank.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

CHAPTER 5 *Unicoi* and Embedded

5.1 Introduction

L-3 Maritime Systems spent time looking at several commercially available source stack vendors. For the Phase 2 we used Telesoft International. Their offering had the benefit of offering several different stacks for both VoIP and ISDN. The Phase 1 recommendation was to have Phase 2 create the Proof-of-Concept with support for VoIP and then Phase 3 would add the BRI hardware and stack to make a complete hybrid solution. The stack was very flexible and could be used on several different platforms. This was both a benefit and a disadvantage; it used `#defines` through the code making tracking and modifying for AS-SIP very difficult. In the initial discussions with SPAWAR they wanted to have a hybrid design that would allow them to ingrate from the current technologies ISDN and BRI to VoIP with the same piece of equipment. During Phase 2 following attendance and presentation at the Internal Communications Symposium put on by SPAWAR, it was determined that the hybrid had less interest than it did in the previous Symposium. The Phase 2 Proof-of-Concept was developed with the Telesoft International SIP stack that AS-SIP was added to and then shown at the JUICE 2009 exercise. During the end of Phase 2 and before Phase 3 began, L-3 Maritime Systems started to look for a stack that was easy to rapidly modify for the changes required for AS-SIP and the nature of this dynamic development process. The first step of the Phase 3 project was to change the direction from a hybrid system to an embedded stack that would allow for a better representation of the feasibility of implementing AS-SIP for the US Navy.

The stack that was selected and incorporated into the Phase 3 solution was from *Unicoi* Systems Incorporated. The material that follows is produced with Consent and is taken from the “InstaVoIP Overview”:

Unicoi Systems, Incorporated

410 Peachtree Parkway
Building 400, Suite 4226
Atlanta, GA 30041
Phone: +1.678.208.2250
Fax: +1.678.208.2254

The following pages of this section are the contents of the overview and show features that give the reader a good understanding of the magnitude of the product, and the need to have a stack that is based on a good flexible base to allow the development to be used on different platforms. In this case we use the embedded source that ran on the *BlackFin* Processor from Analog Devices. We also used the same base SIP stack and compiled it for Windows to run as a soft phone as represented in the GUI section of this report.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

To move forward with this development project with *Unicoi*, the stack needs to be purchased from *Unicoi* sales: sales@unicoi.com. After the purchase of the stack, they supply a good support group through this email address: support@unicoi.com. The source code that is contained in this report and the complementary Embedded Development report (Volume 2), does not contain code from *Unicoi* stack source, but is complementary code that added the AS-SIP into their current product. See the Embedded Development Report for more detailed information on how to implement AS-SIP along side of the *Unicoi* stack.

5.2 Overview

The **InstaVoIP** family adds VoIP capabilities to any product. Available pre-optimized for select platforms or with full source code for porting to any platform, **InstaVoIP** provides a robust VoIP core with a flexible API which allows customization into nearly any end product.

NOTE: InstaVoIP is an evolving product with active, on-going development. Roadmap items are marked with this icon: 🚧; please contact us for more information on the current status of these items.

5.3 InstaVoIP Family

5.3.1 InstaVoIP Embedded

A full ANSI C source code release allows **InstaVoIP** Embedded to be used on any platform. Simply porting the Fusion Common Layer (FCL) abstraction to a platform's RTOS/OS, network stack, and file system (optional), and implementing an audio driver channel for the platform's audio hardware is all that is necessary to start making VoIP calls. Additionally, having full source code allows customer changes to be made to the code (such as implementing newer or less popular RFCs), and aids in debugging low-level problems.

An optimized library release for Windows, Linux, and Mac OS 🚧 desktops allows for rapid integration without dealing with any porting or audio issues. Standalone SoftPhone application examples are provided, or the libraries can be integrated into existing applications.

5.3.2 InstaVoIP Mobile

An optimized library release for iOS, Android, and Windows Mobile 🚧 environments allows for rapid integration without dealing with any porting or audio issues. Standalone SoftPhone application examples are provided, or the libraries can be integrated into existing applications.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

5.3.3 InstaVoIP Module

A hardware/software combination, **InstaVoIP** Module is a standalone hardware module ready to be dropped into an existing hardware platform or used as the foundation of a new product. Based on the powerful Analog Devices *Blackfin* processor, **InstaVoIP** Module includes an Ethernet connection, 4 MB of firmware and file system storage and a 48 kHz capable stereo audio codec. The efficient Fusion RTOS, robust Fusion Network TCP/IP stack, and InstaVoIP Embedded are pre-integrated; out-of-the-box the **InstaVoIP** module powers up and is immediately capable of creating 5-way conference calls using the G.729 codec and while doing so it still has enough free processing power to run user applications.

The software for **InstaVoIP** module is available in two varieties: a combination object/source code release, or a full source code release. In the combination release, the majority of the core code is provided in object format but key sections are provided as source to allow for user expansion. The full source code release is for customers that typically need to modify the core code to implement new functionality, or for those customers who prefer to have the source for debugging purposes.

5.3.4 Example Applications

There are virtually no limits to how **InstaVoIP** can be used to create VoIP-capable applications. Some example applications are:

SoftPhone

InstaVoIP can easily be made into a standalone SoftPhone application. Example projects are included in the Embedded, Desktop, and Mobile versions which provide instant functionality and are ready for customer differentiation.



Desktop Phone

InstaVoIP Embedded or Module can be used to create enterprise-ready desktop phones that interoperate with the most popular PBX solutions in the market.



Call Box

InstaVoIP's User Interface API allows for traditional interfaces as well as highly differentiated user interfaces such as a Call Box with its simpler “press this button in case of emergency” interface. Various aspects of a call workflow can be controlled by a user or controlled by custom code which allows for the implementation of any required UI.





Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

FXS/FXO Analog Gateway

InstaVoIP can be used to create an FXS Gateway (where analog phones plug into the device) or an FXO Gateway (where the device plugs into an existing POTS line) thanks to the FXS/FXO workflows 🚧 included in the InstaVoIP Call Manager.



RoIP

Radio-over-IP is a growing market that gives traditional radios connectivity into corporate PBXs and/or traditional POTS lines. **InstaVoIP** Embedded or Module can be used to quickly create devices that bridge these environments for military, public emergency systems, or businesses that use radios in their daily environment.



Analog-to-Digital Transition

There are many products in the marketplace today that have analog phone interfaces and these products need to be modernized for use VoIP. With **InstaVoIP** you can quickly add VoIP where it can replace the existing analog interface or work alongside it to provide a smooth technology transition.

Add-On VoIP Capabilities

InstaVoIP can be embedded into an existing application to provide VoIP capabilities. Use the Desktop or Mobile versions if integrating into an application on one of the supported platforms, or use the Embedded version for any other platform.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

5.4 Features Overview

InstaVoIP includes a core set of basic features and optional features that enable further customer differentiation:

- Core VoIP Networking Protocols
 - SIP, SDP, RTP, STUN 🚧
 - *Optional*: SIPS 🚧, SRTP 🚧 for secure communications
- Call Management
 - *Supported Workflows*: SoftPhone, Desktop Phone, POTS FXS 🚧, POTS FXO 🚧
 - *Actions*: place calls, answer calls, disconnect calls, on/off hold, transfer, conference, generate DTMF, etc.
 - *Events*: incoming call, peer on/off hold, peer disconnect, being transferred, DTMF received, registered/unregistered, etc.
 - Call management control via HTTP/JSON-based web service for remote control
- Voice engine
 - *Codecs*: G.711, G.726 (16/24/33/40 kbps), G.722, DVI4 (narrow/HD/Ultra HD), Linear PCM
 - *Optional Codecs*: G.729, iLBC, G.723, Lockheed TDVC
 - *Algorithms*: Gain, Automatic Gain Control (AGC), DC Blocker, High-Pass Filter, Voice Activity Detector (VAD), Acoustic Echo Suppressor 🚧, Sample Rate Conversion, DTMF (Generator/Detector), Call Progress Tone Generator, Custom Ring Tone Generator, Comfort Noise Generator, Packet Loss Compensation
 - *Optional Algorithms*: Custom Tone Generator, Acoustic Echo Canceller 🚧, Line Echo Canceller 🚧 (currently Module only), Noise Reduction 🚧, Frequency Equalizer 🚧
- Info Subsystem
 - Configuration Information Management
 - File-based by default
 - Can integrate with platform's configuration style
 - Runtime Information Management (e.g. call status)



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- Access via HTTP/JSON-based web service for remote configuration and status monitoring
- Web-Based Configuration UI
 - *Optional*: HTTPS for secure access
 - Expandable to include user-application configuration 🚧
- Module Only
 - Includes **InstaVoIP** Embedded, plus:
 - Fusion RTOS
 - Fusion Embedded™ TCP/IPv4 Networking Stack, *optional* IPv6 🚧
 - Fusion Embedded File System
 - Hardware
 - BF516 running at 300MHz, 8MB RAM, 4MB Flash
 - SSM2603 high-fidelity stereo audio codec
 - 10/100Mbps Ethernet via RMII
 - SD Card Support 🚧, Digital GPIO
 - All software and hardware already integrated and optimized

5.5 Architecture

The architecture of **InstaVoIP** is shown in Figure 5-1, and explained further in sections below:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

All Fusion Embedded products are written to the FCL porting layer interface, an implementation of which must be available on each platform. Existing ports are available for Fusion Embedded, Win32/CE, Linux (or other POSIX), MacOS, iOS, Android, and uCOS-I/2/3; unsupported example ports are available for select other embedded RTOSs.

See the porting section below for a brief overview of the porting process.

5.5.3 Audio Driver Channel

FCL does not provide a standard abstraction for audio hardware access, so **InstaVoIP** defines its own in the form of an “audio driver channel”. This logical object receives analog data from the microphone and passes it to the Voice Engine; it also gets analog data from the Voice Engine and delivers it to the speakers. How this is done is platform dependent.

InstaVoIP is available with a few Audio Driver Channel implementations:

- A generic implementation using the open-source project PortAudio; this works immediately on platforms where PortAudio has been ported (Win32/CE, Linux, MacOS, etc.).
- An optimized implementation for Android
- An optimized implementation for MacOS and iOS.

For platforms where an existing port is not available, an Audio Driver Channel must be implemented. More details can be found in the *Fusion Embedded Voice Engine User's Guide*, but the general idea is that 10ms chunks of analog audio are put into and taken from voice engine queues and this content is mapped from/to appropriate buffers on the platform.

5.5.4 VoIP Networking Protocols

Full, expandable implementations of core VoIP networking protocols are included and pre-integrated with the Call Manager and Voice Engine. These protocols are:

- *SIP/SDP*: The Session Initiation Protocol (SIP) provides the functionality to register with SIP proxy servers, is used in managing individual calls (connect/disconnect, on/off hold, transfers, etc.), used for managing presence 🚩, and provides event notification 🚩 (such as message-waiting indication). The Session Description Protocol (SDP) is used inside of certain SIP messages to provide details used during the different workflows.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- *RTP*: The Real-time Transport Protocol (RTP) is used to send real-time content (such as audio or out-of-band data like DTMF). The protocol provides metadata used to help receivers deal with network conditions such as jitter and lost packets.
- *IAX*: 🚧 The Inter-Asterisk eXchange (IAX) protocol is an optional protocol used by Asterisk servers which provides similar services to SIP/RTP. This is used for interoperability with Asterisk-based environments configured to use this protocol; the significant majority of new implementations of VoIP will use SIP/RTP.
- *STUN*: 🚧 The Session Traversal Utilities for NAT (STUN) protocol helps SIP/RTP properly transition through Network Address Translation (NAT) modification done by most firewalls. This is important when “external” users need to connect to an “internal” system protected by a firewall.
- *SIPS/SRTP*: 🚧 The Secure SIP (SIPS) and Secure RTP (SRTP) protocols are used when communications must be secure from eavesdropping; to be fully secure both protocols must be used in conjunction with one another. SIPS by itself is also useful for NAT/firewall traversal.

5.5.5 Voice Engine Audio Processing

The Voice Engine Audio Processing system, the core of the Fusion Embedded Voice Engine, implements the workflow which controls the encoding/decoding of audio packets, the application of various algorithms, and the mixing of multiple audio channels.

Analog channels (microphone, speaker, FXO/FXS, etc.) and digital channels (RTP) are implemented with different algorithmic combinations with fine control for each algorithm in the workflow. It has a plug-in architecture for adding additional codecs and echo cancellers (each of which may be available in optimized form on various platforms).

A jitter buffer is included to handle network jitter, packet bursts, and packet loss.

There is also an API to control Call Progress Tones (CPT) and play audio files (e.g. voice prompts for implementing an answering machine).

5.5.6 Call Manager

The Call Manager provides a simplified interface for dealing with standard call workflows such as a softphone, desktop IP phone, an FXS/FXO gateway, etc. The Call Manager hides the complexities and confusion of the SIP/SDP and RTP protocols which were not designed exclusively for VoIP applications. It also provides a consistent interface regardless of the underlying service protocol (i.e. SIP vs. IAX vs. FXO).



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

User Interfaces (UIs) are built using the call workflow API which is a combination of function calls and event notification. With this API you can also handle call transfers and call conferencing without the need to understand the nuances of the underlying protocol.

5.5.7 Info Subsystem

Centralized access to configuration data (which is persisted) and runtime data (which is not persisted) is provided by the Information Subsystem. This includes information such as server account configuration to active call information.

Configuration data is persisted and loaded at startup time; the default implementation uses JSON-based files 🚧 but can be replaced by a platform-appropriate solution.

An expansion API is provided that allows custom user information to be tracked and additional configuration information to be persisted and loaded. 🚧

5.5.8 Configuration Web Application and Web Service

A web-based interface for accessing information subsystem data and managing configuration data is provided both as an HTML-based web application and JSON-based web service.

The web application uses standard HTML/CSS for its design which allows for simple re-skinning, but it also uses the web service as its data channel. This keeps the presentation layer separate from the data layer which allows for complete restructuring of the look and feel without worrying about tightly coupled data access.

The web service also allows for access and configuration via an external application or device.

5.5.9 Call Manager Web Service

Similar to the Configuration Web Service, there is a JSON-based web service providing access to the Call Manager Interface API. This allows control of InstaVoIP from an external application; some customers have used centralized management applications to control banks of **InstaVoIP**-powered devices.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

5.5.10 Expansion APIs

InstaVoIP has expansion APIs in key areas in the architecture which allow great flexibility in creating new or integrating into existing products.

5.5.11 Call Manager Interface

This is the main API that is used to integrate **InstaVoIP** into a product; it is used to instigate actions (place a call, disconnect a call, etc.) and handle asynchronous events (incoming call, peer disconnected, etc.).

As a short generic example of how the API works, let's follow the workflow of an application initiating an outgoing call on a SIP-based service. One thing to realize is that the entire API is event driven because of the need to synchronize actions between various APIs. So every function call returns immediately and at some point in the near future an event is received with the result of the function call.

The first thing an application would do is create a new outgoing-call structure. When this structure is fully initialized by SIP an "outgoing ready" event is received. At this point the application could choose to manipulate the call structure in some way (e.g. add custom headers such as those required by Assured-Services SIP (AS-SIP) 🚧) or just invoke the dial action. When SIP processes this call and the INVITE has been sent a "dialing" event is received.

At this point, events will come in as things happen down in the SIP layer. In normal scenarios, for example, a "ringback" event will come in and when the call is answered an "active" event is received.

5.5.12 Info Subsystem

The Info Subsystem is implemented as a hierarchal data model to which applications can add their own nodes; information can be added as runtime storage or as data that should be persisted. Using the info subsystem is beneficial because it allows use of helper functions that make passing JSON-formatted data to/from web pages very simple.

5.5.13 Configuration Web Application

The Configuration Web Application can be modified or expanded on many levels. Tweaks to CSS allow for quick branding without changing the general style or content, or HTML access allows the addition or removal of data fields or a complete replacement of the default style.

Additional "process functions" can be added to the underlying HTTP server to provide completely new functionality.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Data access/modification is done through the Info Subsystem which has helper functions to go to/from JSON format. Using a naming convention for input field IDs allows JavaScript helper functions to automatically populate fields and submit field values back to the server.

5.5.14 Voice Engine

If a platform has optimized versions of existing or new codecs, they can easily be integrated using the Voice Engine's Codec API. Similarly, the Voice Engine's Echo Cancellation API allows optimized or third party echo cancellation implementations (this includes full-duplex or half-duplex acoustic echo cancellation or G.168-style line echo cancellation).

5.6 Porting

The porting effort for **InstaVoIP** can be broken down into the following areas: FCL, Audio Driver Channel, and Info Subsystem network configuration access.

5.6.1 FCL

The FCL port consists of mapping macros to functions, and if necessary writing wrapper functions to make platform implementations conform to the FCL porting layer interface. This is done for C runtime library functions, RTOS/OS threading and thread synchronization, and networking.

The C runtime library porting is already done for most platforms. On occasion a few specific functions may not be available on a particular platform and in these cases the macro can be defined to use the FCL implementation (FCL comes with a basic C runtime library implementation).

RTOS/OS threading functions are already ported for several different platforms, but on new platforms wrapper functions must be written (and can be modeled after existing ports) to create new threads and manage semaphore and critical section synchronization objects.

The FCL networking interface follows the Berkeley standard, so any stack that follows this standard is already ported. For stacks that do not support this standard, wrapper functions must be written to convert the native API to the FCL interface.

5.6.2 Audio Driver Channel

The Audio Driver Channel port, at a high level, consists of dealing with two queues for each analog audio channel.

The speaker queue gets loaded by the Voice Engine ever 10ms with audio content and it is the channel's responsibility to remove this content (as quickly as the platform will



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

allow) and send it to the platform's audio driver. The microphone queue must be loaded by the channel with 10ms of audio content as efficiently as possible.

Buffering of data must be done in some cases to match a platform's capability to the Voice Engine packets. For example, a speaker driver on a particular platform may take data in buffers of 70ms. In this case, the 10ms packets from the Voice Engine must be buffered into a 70ms buffer before delivery to the hardware driver.

Similarly if a microphone driver provides 70ms of data this must be split into seven 10ms packets and fed into the microphone queue.

An implementation of an Audio Driver Channel for the open-source project PortAudio is provided and can be used as a template for other platforms.

5.7 Info Subsystem

Certain aspects of the VoIP protocol require the network address of the device. In these cases, the VoIP code asks the info subsystem for the device's address and obtaining this information is platform specific.

A device must implement a function that will provide its external network address.



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

This page intentionally left blank.



CHAPTER 6 *Unicoi* Graphical User Interface

6.1 Windows GUI

6.1.1 Introduction

There are currently three different iterations or prototypes of the Windows GUI. Each of them use different amounts of the Fusion codebase. We refer to these prototypes as the Cougar Win32 Net Client, the Headless Client and the Fusion SipPhone. Each of these prototypes has different features and functionality. The Cougar Win32 Net Client is the latest iteration and the Fusion SipPhone is the oldest iteration.

6.1.2 Cougar Win32 Net Client

This is the latest iteration of the Windows GUI and shares the most code with the embedded prototype. Its GUI is built using Windows Forms and Microsoft .Net 2.0 Framework. It also uses WinSock2 for network connections. It is provisioned using the same HTTP interface as the embedded prototype and like the embedded prototype also offers a telnet connection for debugging and logging.

6.1.3 Features

The features it supports are:

- Make / Receive VoIP Calls
- Blind Transfer, Consultant Transfer, Call Hold and Retrieve
- Multiple Call Appearances
- Secure SIP using TLS with both simple and mutual authentication
- Secure RTP and RTCP
- SIP Registration / Authentication
- HTTP Provisioning Interface
- Telnet Shell
- Fusion Call Manager
- Fusion Voice Engine
- WinSock2
- Static or Dynamic IPv4 Dressing
- Windows GUI
- IPv6 SIP Signaling
- IPv6 Auto-addressing



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

6.1.4 Compilation

The project delivered from *Unicoi* was developed using Microsoft Visual Studio 2005. Since L-3 Maritime Systems uses a later version (Visual Studio 2008) the solution and included projects are immediately updated. In order to successfully build the project a few environment variables need to be set. FUSION_DIR needs to be set to the root directory of the fusion source, this is the directory that contains the sub-directories of fusion packages such as {algorithms, common, config, customer, docs, drivers, file, net, ...}

FUSION_LIBRARY_DIR needs to be set to the directory that contains the portaudio library.

Once the solution has been compiled the program can be run and configured.

By running the executable cougar.exe or launching the debugger from inside Visual Studio, the following window will be presented. Notice that its state is **Registering** (Figure 6-1).



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

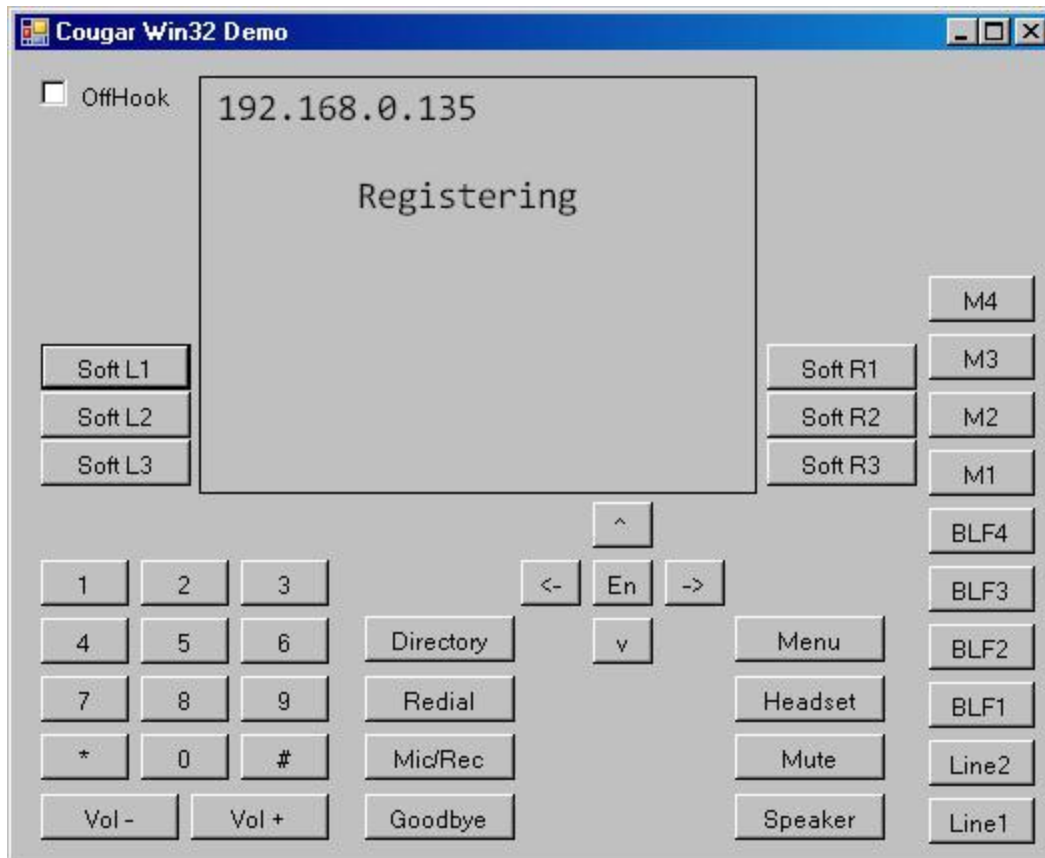


Figure 6-1 Unprovisioned Cougar Win32 Demo

6.2 Configuration

The configuration of the Cougar Win32 Net Client is accomplished identically to that of the Cougar Embedded Prototype. Type the URI of the host running the Client into the address field of a web browser. The HTTP server built into the client will respond to the request asking for a username and password. After the administrators credentials have been verified, a web page similar to the following will be rendered.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

6.2.1 Provisioning Display

Unicoi VoIP Phone Configuration		
Network Status		
Status	Network	
○ System	Address	0.0.0.0
Network Setup	Gateway	0.0.0.0
○ Network	DNS Primary	0.0.0.0
VoIP Setup	DNS Secondary	0.0.0.0
○ Account	DNS Tertiary	0.0.0.0
○ Media	Phone	
○ Security	User	
○ Advanced	Registration Status	Not Configured
System	STUN	Disabled
○ Administration	copyright © 2010 Unicoi Systems, Inc.	
○ Date/Time		
○ Backup/Restore		
○ Upgrade Firmware		
○ Logout		

Figure 6-2 Initial Provisioning Display

6.2.1.1 Network Setup

The first step is to check the Network settings. This is done by clicking the Network link on the left side of the page. When this is selected the following will be rendered.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot displays the 'Unicoi VoIP Phone Configuration' web interface. The browser window title is 'Unicoi VoIP Phone Configuration - Mozilla Firefox'. The page has a dark blue header with the title 'Unicoi VoIP Phone Configuration' and a sub-header 'WAN setup'. On the left, a sidebar menu lists various configuration categories: Status, Network Setup (highlighted), VoIP Setup, and System. The main content area is divided into sections: 'General' with fields for Host (Cougar), Domain (Cougar), and Connection Type (Dynamic IP selected); 'Additional Settings' with MTU Size (1500); and 'VLAN' with a checkbox for 'Enabled' (unchecked), ID (4), and User Priority (0 - Best Effort). A 'Save Changes' button is at the bottom right. The footer contains the copyright notice 'copyright © 2010 Unicoi Systems, Inc.'.

Figure 6-3 Provisioning Network Setup

6.2.1.2 Static Ipv4 Addressing

The default settings assume the use of DHCP but not VLAN. If static IPv4 addressing is to be used then the Static IP field is checked and the following web page is rendered.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'Unicoi VoIP Phone Configuration' web interface in a Mozilla Firefox browser. The left sidebar contains a navigation menu with the following items: Status, Network Setup, VoIP Setup, Account, Media, Security, Advanced, System, Administration, Date/Time, Backup/Restore, Upgrade Firmware, and Logout. The 'Network Setup' section is expanded, showing 'Network' and 'VoIP Setup' sub-sections. The 'WAN setup' page is displayed, featuring a 'General' tab. Under 'General', the 'Host' is set to 'Cougar' and the 'Domain' is also 'Cougar'. The 'Connection Type' is set to 'Static IP'. Below this, the 'Static IP Address' section contains fields for 'Address' (0.0.0.0), 'Mask' (255.255.255.0), 'Default Router' (0.0.0.0), 'DNS Primary' (0.0.0.0), 'DNS Secondary' (0.0.0.0), and 'DNS Tertiary' (0.0.0.0). The 'Additional Settings' section shows 'MTU Size (advanced)' set to 1500. The 'VLAN' section has 'VLAN' checked as 'Enabled', 'ID' set to 4 (with a note '(value: 0 to 4094)'), and 'User Priority' set to '0 - Best Effort' (with a note '(default: 0)'). A 'Save Changes' button is located at the bottom right of the configuration area. The footer of the page reads 'copyright © 2010 Unicoi Systems, Inc.'

Figure 6-4 Static IPv4 Addressing

6.2.1.3 VoIP Account Setup

The first place that data must be entered by the user is in the VoIP Account. This is accessed by clicking the Account link on the left of the page (Figure 6-5). The data entered on this page is the same that will be setup on the VoIP switch. The Username/Number needs to be provisioned on the switch so that registration can occur. The Domain, Registrar and Registrar port are the network location of the SIP Registration Server and also need to be filled in. Auto-configure is unchecked and then the host and well known port of the registrar can be entered. The remainder of the fields are usually not necessary in order to obtain successful registration. This is of course dependent on the switch configuration.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Unicoi VoIP Phone Configuration - Mozilla Firefox

Unicoi VoIP Phone Configuration

Unicoi VoIP Phone Configuration

VoIP Account

Status

- System
- Network Setup
- Network
- VoIP Setup
- Account
- Media
- Security
- Advanced

System

- Administration
- Date/Time
- Backup/Restore
- Upgrade Firmware
- Logout

SIP Configuration

Username/Number	<input type="text"/>
Display Name	<input type="text"/>
Domain	<input type="text"/>
Registrar	<input type="text" value="sip."/> <input checked="" type="checkbox"/> auto-configure
Registrar Port	<input type="text" value="0"/> (advanced; set to 0 for auto detect)

Additional Settings

Outbound Proxy	<input type="text"/> (leave blank if same as registrar)
Outbound Proxy Port	<input type="text" value="0"/> (advanced; set to 0 for auto detect)
Registration Lifetime	<input type="text" value="3600"/> seconds
Keep-Alive	<input type="text" value="Options Request"/> (advanced)
STUN	<input type="checkbox"/> Enabled

Proxy Authentication

Username	<input type="text"/>
Password	<input type="password" value="....."/>

VLAN User Priorities

SIP	<input type="text" value="0 - Best Effort"/> (default: 0)
RTP Audio	<input type="text" value="6 - Voice < 10ms latency and jitter"/> (default: 6)
RTP Text	<input type="text" value="6 - Voice < 10ms latency and jitter"/> (default: 6)

Save Changes

copyright © 2010 Unicoi Systems, Inc.

Figure 6-5 VoIP Account Settings



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Figure 6-6 Minimum VoIP Account Settings

Figure 6-6 shows the minimum amount of data necessary to complete SIP Registration.

6.2.1.4 Media Setup

The default VoIP Media settings are usually sufficient to get started.



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Figure 6-7 VoIP Media Settings

6.2.1.5 Security Setup

The default security settings do not need to be changed for a non-secure call.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Unicoi VoIP Phone Configuration - Mozilla Firefox

Unicoi VoIP Phone Configuration

Unicoi VoIP Phone Configuration

VoIP Security

Status

- System
- Network Setup
- VoIP Setup
 - Account
 - Media
 - Security**
 - Advanced
- System
 - Administration
 - Date/Time
 - Backup/Restore
 - Upgrade Firmware
 - Logout

Negotiation Options

SIPS Security	Disabled	(default: Disabled)
SRTTP Security	Disabled	(default: Disabled)
SRTTP Encryption	Prefer OFF	(default: Prefer OFF)
SRTTP Authentication	Prefer OFF	(default: Prefer OFF)
SRTCP Encryption	Prefer OFF	(default: Prefer OFF)

Advanced Options

MKI	<input type="checkbox"/> Enabled
Key Lifetime	0 packets (min: 1024; 0 for max)

SRTTP Crypto Suite Selection

Available		Preferred
AES_CM_128_HMAC_SHA1_	Add >> Remove <<	AES_CM_128_HMAC_SHA1_
AES_CM_128_HMAC_SHA1_		AES_CM_128_HMAC_SHA1_
		Move Up Move Down

Save Changes

Figure 6-8 Security Settings

6.2.1.6 Advanced Setup

The default advanced settings are adequate to get started.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Figure 6-9 Advanced Settings

6.2.1.7 Persistent Data

The Cougar Win32 Demo software will write the settings to the `.\config\voip.xml` file whenever the Save Changes button is clicked. There are other elements in this file that are set to their defaults when this file is created. The configuration file (Figure 6-10) can be viewed with a browser or other XML editor/viewer.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
<?xml version='1.0'?>
<fcm-voip version='4.2'>
  <fcm-sip enabled='1'>
    <account id='1' label="" enabled='true'>
      <sip user='user4003' name="" stun='0' rtt='1' dscp='0'>
        <proxy domain='192.168.0.11' registration-lifetime='3600' keep-alive='none'>
          <registrar address='192.168.0.11' port='5060' auto-registrar='0' />
          <outbound-proxy address="" port='0' />
          <authentication username="" password="" />
        </proxy>
        <session-timers enabled='false' min='60' default='90' max='300' refresher='none' />
      </sip>
      <rtp base-port='23456' dscp='46' />
      <security>
        <sips security='0' />
        <srtp security='0' encryption='1' authentication='1' srtcp-encryption='1' mki='0' key-lifetime='0d'>
          <crypto-suites>
            <suite id='1' />
            <suite id='2' />
          </crypto-suites>
        </srtp>
      </security>
      <vlan-user-priorities sip='0' rtp-audio='6' rtp-text='6' />
    </account>
    <certificate preference='1' location='./certs/' />
    <stun server="" port='3478' />
  </fcm-sip>
  <fcm-iax enabled='1'>
    <account id='1' label="" enabled='6072432'>
      <iax user="" name="">
        <proxy domain="" registration-lifetime='3600'>
          <registrar address="" port='0' auto-registrar='1' />
          <authentication username="" password="" />
        </proxy>
      </iax>
    </account>
  </fcm-iax>
</fcm-voip>
```

Figure 6-10 VoIP XML Persistent Data

6.2.1.8 Execution

This section describes some of the basic uses of the Cougar Win32 Demo.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

6.2.1.9 Registration

After the Account Settings have been set properly the client will attempt to register with the registrar using the Username / Number, Registrar and Registrar Port that have been provisioned.

The client, when successfully registered will display the window shown in Figure 6-11. This is very similar to the layout of the embedded device with 6 soft keys around the display and the same 10 keys along the right hand side of the device.

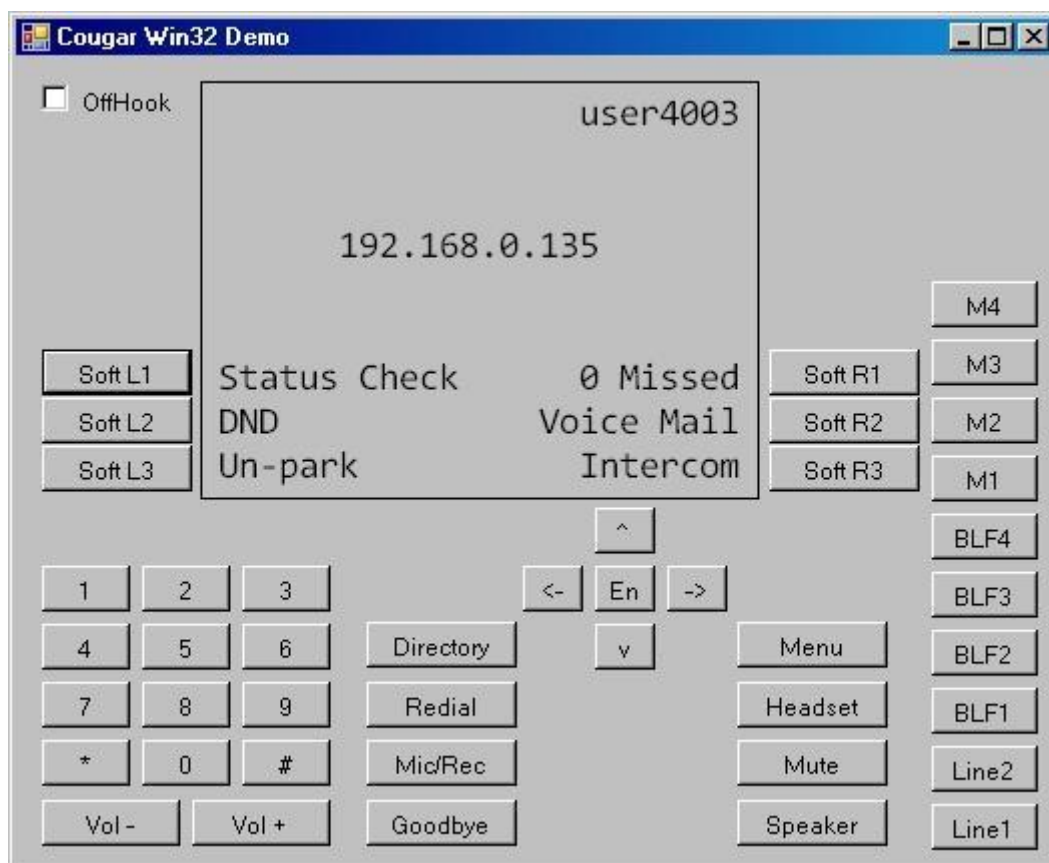


Figure 6-11 Cougar Win32 Demo Registered



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Here are the SIP Messages captured and displayed (Figure 6-12) by the Wireshark Protocol Analyzer. This SIP transaction contains only a Registration Request and a Status Reply, in this case 200 which signifies that registration was successful.

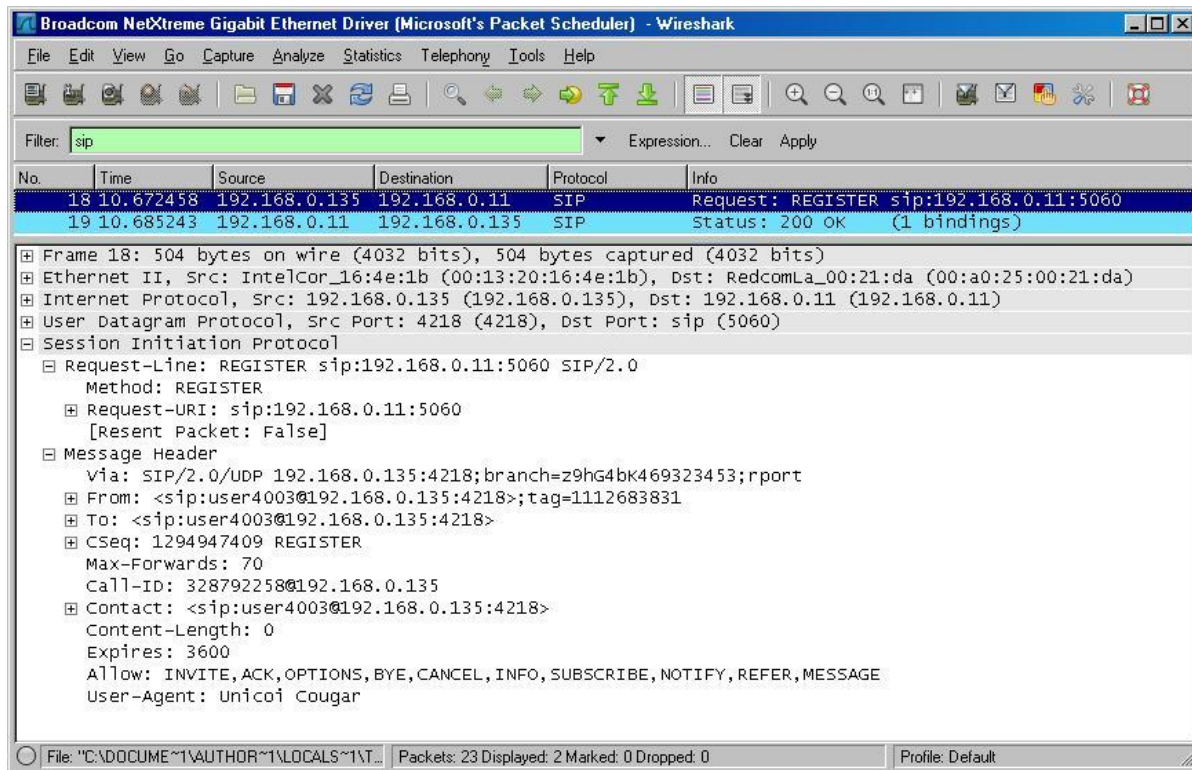


Figure 6-12 Wireshark – SIP Register

After the Cougar Win32 Demo software has registered it is allowed by the switch to make and receive calls. To make a call the Demo user clicks the icons on the phone to emulate key touches on the embedded device. For example, this window has had the icons 5, 5, 5, 3, 0, 0, 2 clicked. Each click causes the window to be updated with the data selected.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

6.3 Making a SIP Call using Cougar Win32 Demo

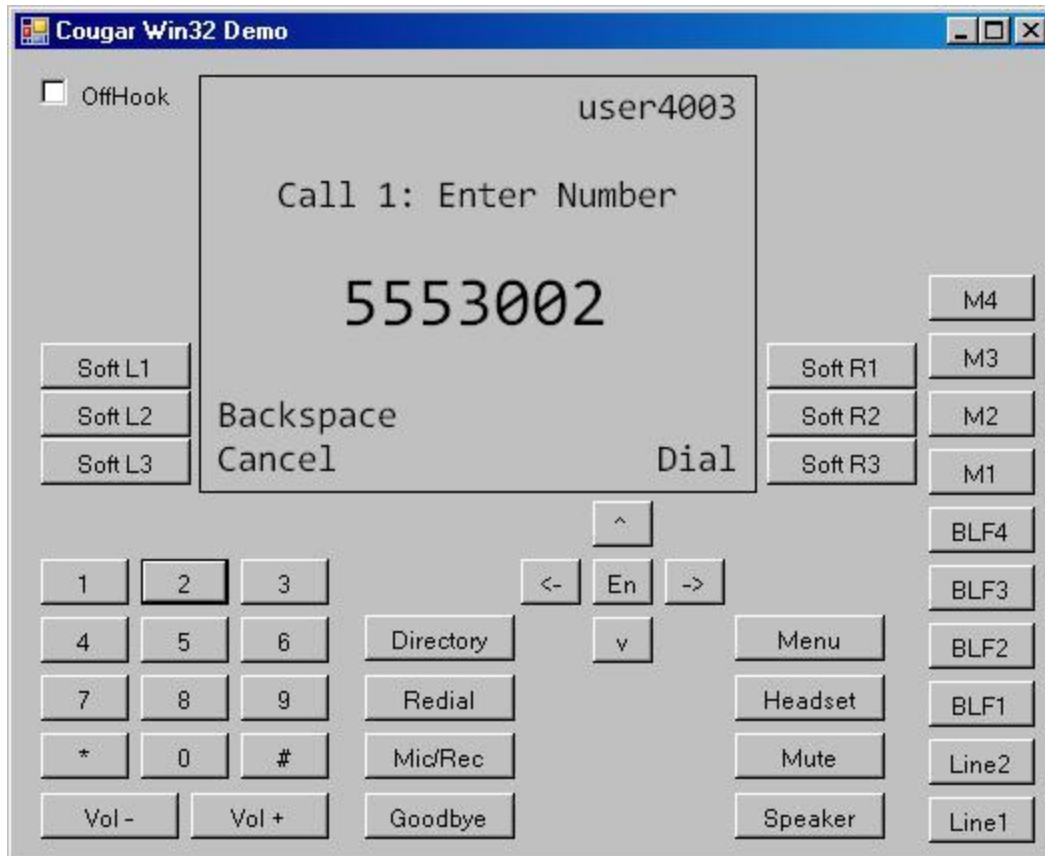


Figure 6-13 Cougar Win32 Demo – Make Call

After the desired number is entered the Soft R3 icon is clicked to dial the call. When The Dial icon is clicked the display will change and the Windows Playback Device will receive and play ringback audio.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

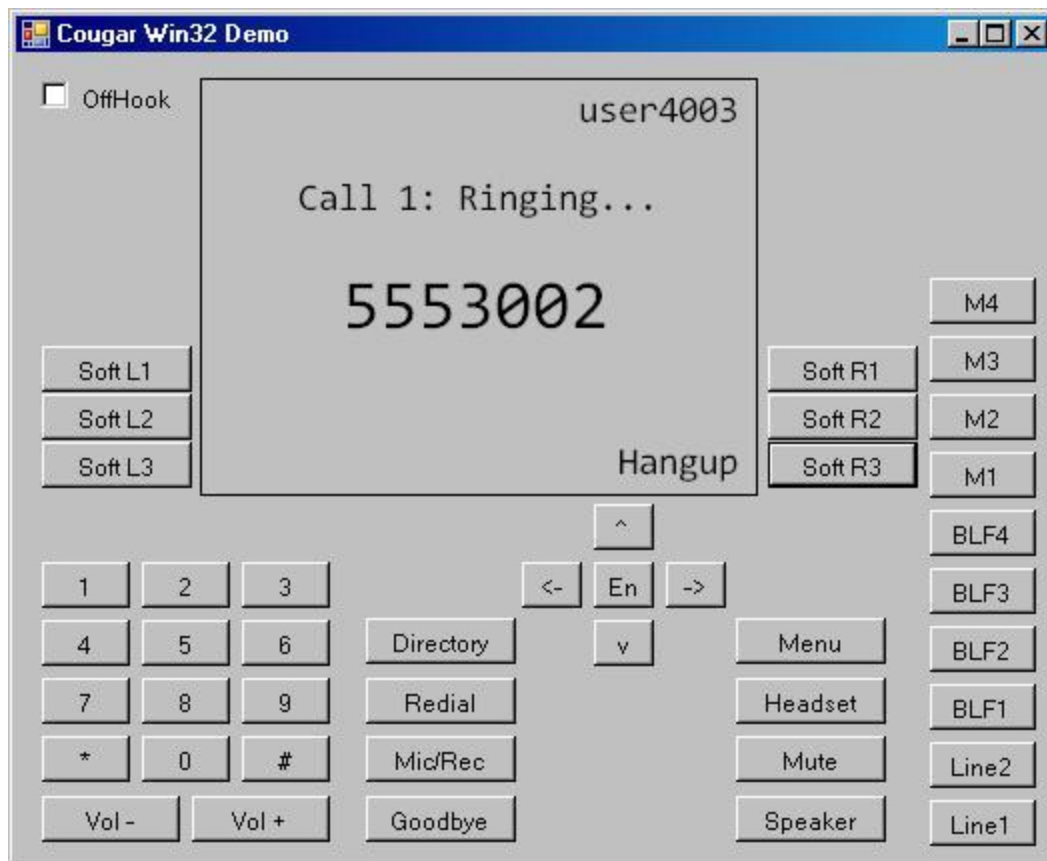


Figure 6-14 Cougar Win32 Demo – Ring-back

6.3.1 Active 2 Party SIP Call

When the called party answers the ringback audio will be replaced with the far end audio stream and the near end audio will be captured from the Windows recording device and sent to the far end. Here is a Wireshark protocol capture of the SIP Messages between the calling party (user4003), the switch (192.168.0.11), and the called party (555-3002). The called party answers the call after 2.8 seconds. After about 4 seconds the called party hangs up and both sides are disconnected.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

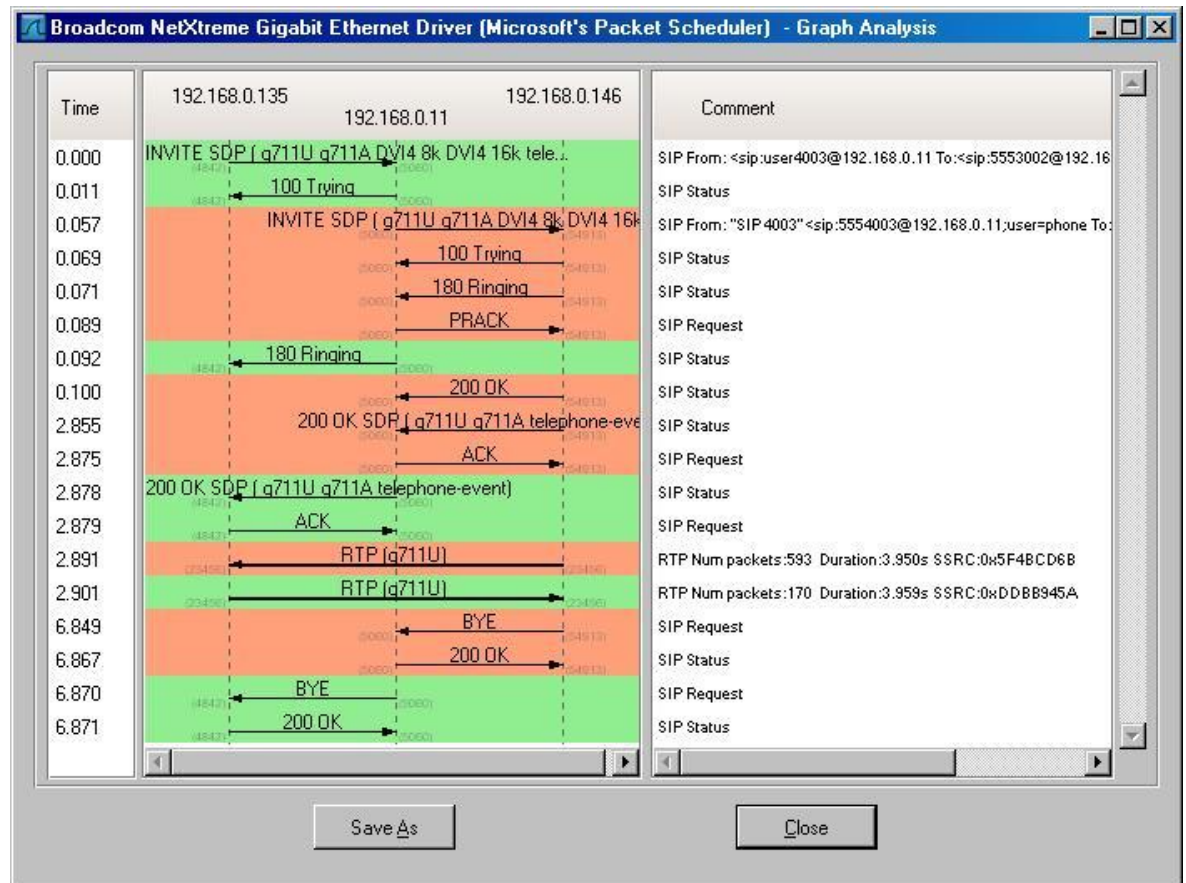


Figure 6-15 Wireshark VoIP Telephony Analysis – SIP Call

Figure 6-15 is another view of the call using the Wireshark Telephony Analyzer which shows the RTP traffic and gives an easier to understand view of how the switch operates between the two parties.

6.3.2 Summary

6.3.2.1 Pros

- Uses most of the same code as the Embedded Client including the Call Manager, Voice Engine and GUI Framework.
- Can operate securely using SIP over TLS and Secure RTP.
- Same provisioning model as Embedded Client
- Uses state-of-the-art Interactive Development Environment (IDE).



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

6.3.2.2 Cons

- No Integration with IPv6
- Relies on WinSock2 and Microsoft .Net Framework 2.0

6.4 Headless Client

The headless client is a Fusion Cougar SIP Client that contains the Fusion Call Manager and Fusion Voice Engine but not any type of native graphical user interface. The software does have an HTTP interface and is built to run on the Windows platform. This section will describe in detail its features and operation. The Headless Softphone is sometimes referred to as the Command Phone.

6.4.1 Features

The features it supports are:

- Make / Receive VoIP Calls
- Blind Transfer, Consultant Transfer, Call Hold and Retrieve
- Multiple Call Appearances
- Secure SIP using TLS with both simple and mutual authentication
- Secure RTP and RTCP
- SIP Registration / Authentication
- HTTP Provisioning Interface
- Telnet Shell
- Fusion Call Manager
- Fusion Voice Engine
- WinSock2
- Static or Dynamic IPv4 Dressing

6.4.2 Compilation

The Headless Client is built using the cmdPhone solution. This was delivered from *Unicoi* as a Microsoft Visual Studio 2005 Solution. Since Visual Studio 2008 is being used for Windows prototypes the solution was updated using the wizard provided by Microsoft for updating Visual Studio Solutions and Projects from 2005 to 2008. No source file changes were required.

The solution contains only one project. That project then contains the many source files distributed amongst the folders that group similar functionality together. For example all of the RTP related source code resides in the **rtp** sub-folder under the **net** folder.



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

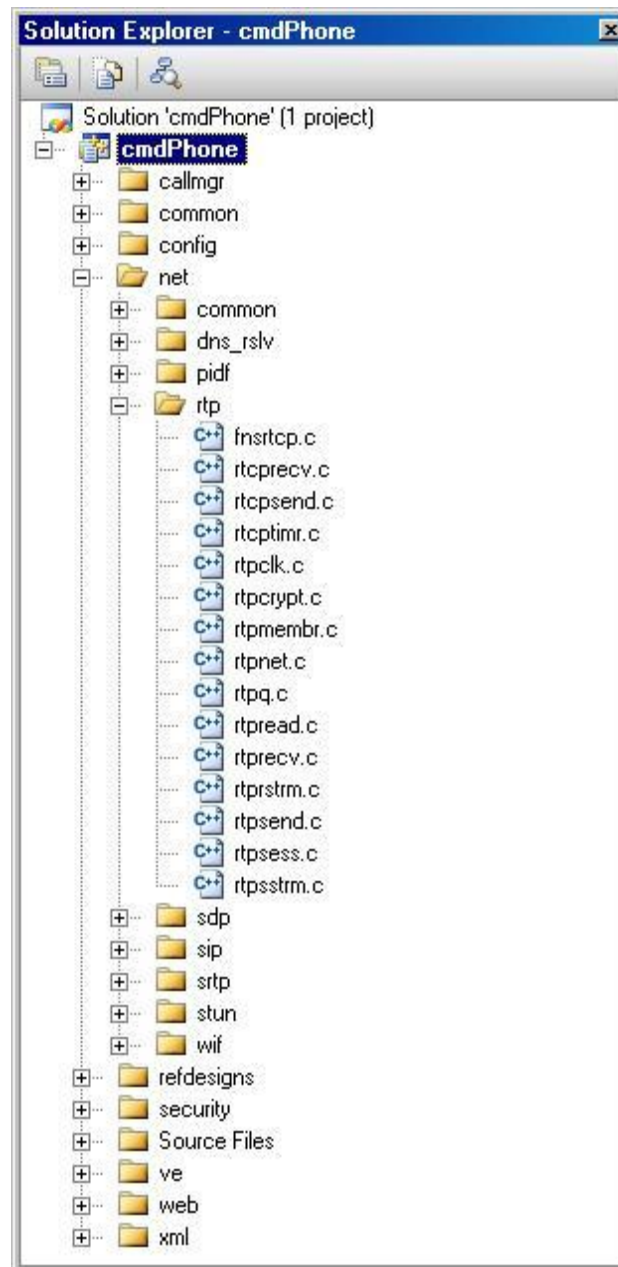


Figure 6-16 Visual Studio – Headless Client Solution

6.4.3 Configuration

Configuring the Headless Client is the exact same as the Fusion Cougar Win32 Demo software and the Embedded Fusion Cougar software. Open a browser window to the host and port that is running the software is listening to. The default well known port is 8080.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The data is stored in the current directory in the voip.xml file which can be identical to the voip.xml file used in the Fusion Cougar Win32 Demo software.

6.4.4 Execution

When the solution is correctly built an executable image for Windows named cmdPhone.exe will be produced. Just like the Fusion Cougar Win32 Demo it will create the XML files that contain the default values if they do not exist. The user can then use a web browser to fill in the appropriate settings.

When the software is run, it will display a console that provides feedback about the events received and the state changes that the software encounters. For example this console window shows that the Headless Client has successfully registered and initialized all of its allowed 6 call appearances to idle.

```
C:\Projects\catamount\myfusion\refdesigns\voip\ip_phone\ports\cmd\win32\Debug\cmdPhone.exe
FCM: Line 0 Call 4: Event Received: INT_SIP_CONNECTED; Arg 0: ' '; Arg 1: ' '
FCM: Handle INT_SIP_CONNECTED on [00:04] SIP_CONFIGURED -> IDLE
FCM: fcmHandleIdle
FCM: Current State:
FCM:   Line 00: Active call: 7fffffff
FCM:       Call 00 state: IDLE
FCM:       Call 01 state: IDLE
FCM:       Call 02 state: IDLE
FCM:       Call 03 state: IDLE
FCM:       Call 04 state: IDLE
FCM:       Call 05 state: SIP_CONFIGURED
FCM: Waiting For Event...
FCM: Line 0 Call 5: Event Received: INT_SIP_CONNECTED; Arg 0: ' '; Arg 1: ' '
FCM: Handle INT_SIP_CONNECTED on [00:05] SIP_CONFIGURED -> IDLE
FCM: fcmHandleIdle
FCM: Current State:
FCM:   Line 00: Active call: 7fffffff
FCM:       Call 00 state: IDLE
FCM:       Call 01 state: IDLE
FCM:       Call 02 state: IDLE
FCM:       Call 03 state: IDLE
FCM:       Call 04 state: IDLE
FCM:       Call 05 state: IDLE
FCM: Waiting For Event...
```

Figure 6-17 Headless Client – SIP Configured

This console display shows the software with one active call.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

```
G:\ c:\Projects\catamount\myfusion\refdesigns\voip\ip_phone\ports\cmd\win32\Release\cmdPhone.exe
FCM: Callback VE -> fcmVeStateChange [P00:C00] RINGBACK to IDLE
FCM: Line 0 Call 0: Event Transmitted: SIP_ACTIVATE; Arg 0: ' '; Arg 1: ' '
winMME AbortStream: waiting for background thread.
FCM: Callback VE -> fcmCallbackVeInitBuffers [P00:C00]
FCM: AM -> fcmDsmTelephoneEventIsSupported [P00:C00]
FCM: AM -> fcmDsmGetPreferredCodec [P00:C00]
FCM: Callback VE -> fcmVeStateChange [P00:C00] IDLE to ACTIVE
FCM: AM -> fcmDsmGetPreferredCodec [P00:C00]
Pa_StartStream: waveInStart returned = 0x0.
FCM: Callback UI -> fcmCallbackUiActivate [P00:C00]
FCM: Callback UI -> fcmCallbackUiAnswered [P00:C00]
FCM: Current State:
FCM:   Line 00: Active call: 00000000
FCM:       Call 00 state: ACTIVE
FCM:       Call 01 state: IDLE
FCM:       Call 02 state: IDLE
FCM:       Call 03 state: IDLE
FCM:       Call 04 state: IDLE
FCM:       Call 05 state: IDLE
FCM: Waiting For Event...
EVENT: {"id": 28, "type": "activate", "callId": 0}
EVENT: {"id": 29, "type": "answered", "callId": 0}
```

Figure 6-18 Headless Client – Active Call

The Headless Client has a fairly primitive HTTP user interface. Here it is shown in the same state as the console above, that is, with one active call to 555-3002.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

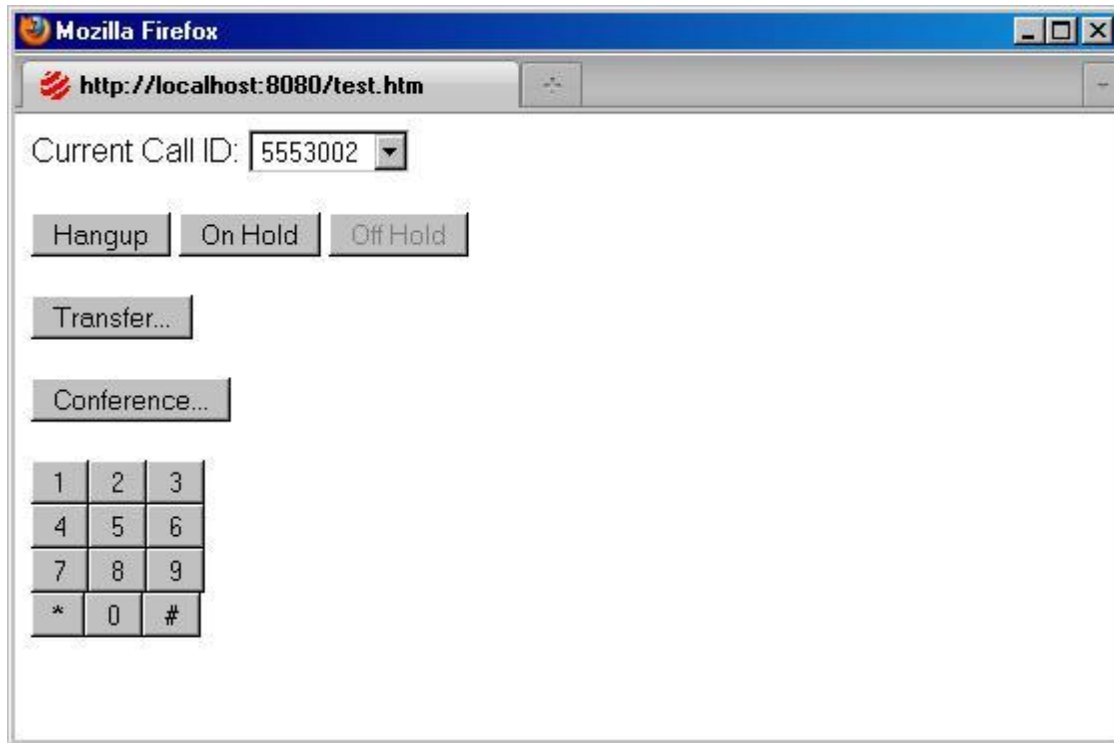


Figure 6-19 Headless Client – GUI Active Call

The main benefit of the Headless Softphone is its HTTP interface. This can be exercised with a command line HTTP client that takes commands and executes them using the HTTP interface.

6.4.5 Media

Regardless of the User Interface that is used to access the headless client to make and receive calls, the media endpoint for the headless client will always be the host that the headless client is executing on.

For example, it is possible to connect to a Windows PC running the headless client from another PC using HTTP. Using this connection the second PC is able to originate, terminate and control call appearances. However, the outgoing audio stream and the incoming audio stream will respectfully originate and terminate at the host running the headless client software.

6.4.6 HttpCmd Utility

The HttpCmd utility is a windows command-line HTTP client that can be used to exercise the Headless Client's HTTP interface locally or from a remote host.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

1. Usage:

usage: httpcmd [<options>] <host-path(s)> [-h <host-path(s)> ...]
host-path(s) : (<host[:port]> <path> | <url>) [[-x] <path> ...]
Options (append all together, e.g. "-gfs")
-g : enable accepting gzip response
-k : enable cookie manager (no persistence)
-p : print to console
-f : save to file
-s : show statistics
-v : verbose (doesn't currently do much...)
Sequence Modifiers
-h : Change host (previous connection closes)
-x : Sync (wait for all queued requests to complete)

Examples:

Simple load of Google Home Page:

httpCmd www.google.com /

or: httpCmd http://www.google.com/

Enable GZIP encoding and save the HTML file:

NOTE: File is saved to default.htm as there is no name specified

httpCmd -gf www.google.com /

Save the Google logo and show statistics:

httpCmd -fs www.google.com /intl/en_ALL/images/logo.gif

Send two requests to Google, retrieve the home page and the logo:

httpCmd -g www.google.com / /intl/en_ALL/images/logo.gif

Send two requests to Google, but disable pipelining:

httpCmd -g www.google.com / -x /intl/en_ALL/images/logo.gif

2. Testing the Headless Client with HttpCmd

For the examples below we use an address of 192.168.0.135 and well known port 8080 for the Headless Client EI. The responses from the HttpCmd will be formatted in JavaScript Object Notation (JSON) [ECMA-262]

In Microsoft Windows XP, the command shell will interpret the ampersand (“&”) as a special character. It is therefore necessary to “escape” the ampersand with another special character, the caret (“^”). This can be seen in the following examples.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

3. Dial a Number

```
C:> HttpCmd -p 192.168.0.135:8080 /cm/dial?poolId=0^&uri=5555002
```

The JSON response:

```
{"callId": 0}
```

The command will optionally reply with Debug information also.

```
HTTPC: Connected : 192.168.0.135:8080
```

```
HTTPC: --> : 192.168.0.135:8080 /cm/dial?poolId=0&uri=5553002
```

```
HTTPC: <-- : 192.168.0.135:8080 /cm/dial?poolId=0&uri=5553002
```

```
HTTPC: Closed : 192.168.0.135:8080
```

4. Hangup an Active Call

```
C:> HttpCmd -p 192.168.0.135:8080 /cm/disconnect?poolId=0^&callId=0
```

5. Send DTMF

```
C:> HttpCmd 192.168.0.135:8080  
/cm/dtmf?poolId=0^&callId=0^&key=<key>
```

Injects an out of band Dual-tone multi-frequency tone. Key can be a number 0-9, *, #, or letters A-D.

6. Accept Incoming Call

```
C:> HttpCmd 192.168.0.135:8080 /cm/accept?poolId=0^&callId=0
```

This command will connect an incoming call.

7. Reject Incoming Call

```
C:> HttpCmd 192.168.0.135:8080 /cm/reject?poolId=0^&callId=0
```

This command has the effect of sending a “busy” to the peer.

8. Go On Hold

```
C:> HttpCmd 192.168.0.135:8080 /cm/goOnHold?poolId=0^&callId=0
```

9. Go Off Hold

```
C:> HttpCmd 192.168.0.135:8080 /cm/goOffHold?poolId=0^&callId=0
```



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

10. Transfer to URI

```
C:> HttpCmd 192.168.0.135:8080  
/cm/xferToUri?poolId=0^&callId=0^&uri=<uri>
```

Transfers the given callId to the given uri. Once successfully transferred the current call is ended. This implements “unattended or blind transfer”.

11. Transfer with Replaces

```
C:> HttpCmd 192.168.0.135:8080  
cm/xferWithReplaces?PoolId=0^&callId=0^&tCallId=<tCallId>^&uri=<uri>
```

Transfers the given tCallId to the given callId. The tCallId call must be on hold when this command is invoked. This implements an “attended transfer”.

12. Add Call to Conference Call

```
C:> HttpCmd 192.168.0.135:8080 /cm/confAdd?poolId=0^&callId=0
```

Adds the given callId, which must be a call on hold, to the currently active call creating a conference call.

13. Remove Call from Conference Call

```
C:> HttpCmd 192.168.0.135:8080 /cm/confRemove?poolId=0^&callId=0
```

Removes the given callId from whatever conference call it is part of. Upon being removed, the call is placed on hold.

6.5 PortAudio Library

The PortAudio Library is an open source library used for processing audio input and output. It is linked into the executable image of both the Fusion Cougar Win32 Demo and Fusion Cougar Headless Client software.

PortAudio is a portable and mature C API for accessing audio hardware. It offers both callback-based and blocking style input and output, deals with sample data format conversions, dithering and much more. There are a large number of implementations available for various platforms including Windows MME, Windows DirectX, Windows and MacOS (Classic) ASIO, MacOS Classic SoundManager, MacOS X CoreAudio, OSS (Linux), Linux ALSA, JACK (MacOS X and Linux) and SGI Irix AL. Because PortAudio has a C API, it can easily be called from a variety of other programming languages.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

From <http://www.portaudio.com>

“PortAudio is a free, cross platform, open-source, audio I/O library. It lets you write simple audio programs in 'C' or C++ that will compile and run on many platforms including Windows, Macintosh OS X, Unix (OSS/ALSA), SGI, and BeOS. PortAudio is intended to promote the exchange of audio software between developers on different platforms[PACOM]¹².”

PortAudio provides a very simple API for recording and/or playing sound using a simple callback function. Example programs are included that synthesize sine waves and pink noise, perform fuzz distortion on a guitar, list available audio devices, etc.” [PACOM]¹³

6.5.1 License

The licensing of the software is a plain MIT license.

“PortAudio Portable Real-Time Audio Library

Copyright (c) 1999-2000 Ross Bencina and Phil Burk

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

* The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ON INFRINGEMENT.

IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The text above constitutes the entire PortAudio license; however, the PortAudio community also makes the following non-binding requests:

* Any person wishing to distribute modifications to the Software is requested to send the modifications to the original developer so that they can be incorporated into the canonical version. It is also requested that these non-binding requests be included along with the license above.”[PACOM]¹⁴

¹² Bencina R., Burk, P., “PortAudio – portable cross platform Audio API”, December 2009.

¹³ Bencina R., Burk, P., “PortAudio – portable cross platform Audio API”, December 2009.

¹⁴ Bencina R., Burk, P., “PortAudio – portable cross platform Audio API”, December 2009.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

6.5.2 Overview of PortAudio

“PortAudio is a cross-platform, open-source C language library for real-time audio input and output. The library provides functions that allow your software to acquire and output real-time audio streams from your computer's sound card or audio interface. It is designed to simplify writing cross-platform audio applications, and also to simplify the development of audio software in general by hiding the complexities of dealing directly with each native audio API. PortAudio is used to implement sound recording, editing and mixing applications, software synthesizers, effects processors, music players, Internet telephony applications, software-defined radios and more. Supported platforms include Microsoft Windows®, Mac OS X and Linux. In addition to the C language, third-party language bindings make it possible to call PortAudio from other programming languages including C++, C#, Python, PureBasic, FreePascal and Lazarus.

Among the operating systems and audio subsystems supported by PortAudio:

- Windows { MME, DirectSound, WASAPI, ASIO }
- Mac OS { Core Audio }
- Linux { ALSA, JACK, OSS }

Below are the steps to writing a PortAudio application:

- Write a callback function that will be called by PortAudio when audio processing is needed.
- Initialize the PA library and open a stream for audio I/O.
- Start the stream. Your callback function will now be called repeatedly by PA in the background.
- In your callback you can read audio data from the inputBuffer and/or write data to the outputBuffer.
- Stop the stream by returning 1 from your callback, or by calling a stop function.
- Close the stream and terminate the library.

In addition to this "Callback" architecture, V19 also supports a "Blocking I/O" model which uses read and write calls which may be more familiar to non-audio programmers. Note that at this time, not all APIs support this functionality. [PACOM]¹⁵

6.5.2.1 Example

Here is a code snippet of “C” source code to open stream using default values.

```
#define SAMPLE_RATE (44100)
static paTestData data;
.....
PaStream *stream;
```

¹⁵ Bencina R., Burk, P., “PortAudio – portable cross platform Audio API”, December 2009.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

PaError err;

```
/* Open an audio I/O stream. */
err = Pa_OpenDefaultStream( &stream,
    0,          /* no input channels */
    2,          /* stereo output */
    paFloat32, /* 32 bit floating point output */
    SAMPLE_RATE,
    256,        /* frames per buffer, i.e. the number
                  of sample frames that PortAudio will
                  request from the callback. Many apps
                  may want to use
                  paFramesPerBufferUnspecified, which
                  tells PortAudio to pick the best,
                  possibly changing, buffer size.*/
    patestCallback, /* this is your callback function */
    &data ); /*This is a pointer that will be passed to
              your callback*/

if( err != paNoError ) goto error;
```

The above example opens the stream for writing, which is sufficient for playback. It is also possible to open a stream for reading, to do recording, or both reading and writing, for simultaneous recording and playback or even real-time audio processing. [PACOM]¹⁶

6.6 Summary

The Headless Client's biggest benefit is the ability to be scripted. It would be fairly easy to set up test scripts that could automate making and receiving calls and rigorously testing the Fusion Cougar software. The tests could be used for memory usage, stability, SIP message processing and regression.

However, the headless client is not particularly useful when testing media. To perform end-to-end tests with media would require other hardware or software to play and record audio into the host computer's devices.

If the headless client had the ability to stream audio from a source other than the default recording device and the ability to sink audio to a source other than the default playing device then the software could be used to test multiple clients on a single host computer which would allow for load testing of the LSC (switch).

¹⁶ Bencina R., Burk, P., "PortAudio – portable cross platform Audio API", December 2009.



CHAPTER 7 Network VoIP Performance

7.1 Introduction

This chapter discusses the testing of the quality of the speech between phones. In the original layout of this project, there were plans to do extensive testing in multiple labs. When we made the change to embedded, and we started to work with the engineering staff at REDCOM Laboratories, Inc., it was determined that concentration on working with them only would move the feasibility study forward quicker. They were in the end of the first JITC round of certification, so we were testing with the Defense Switched Network that is now on the APL (see news release: “REDCOM products dominate the Defense Switched Network Approved Products List” at the following <http://www.REDCOM.com/news.php?id=44>.) Because of the change in direction and focus of the study, limited network affect testing was done. This document is created from the work originally done to prepare the individuals to do the testing in the labs, and to validate the network quality before we started to evaluate the phones capability.

7.2 Voice Quality Testing

This paragraph details the setup, implementation, and execution of testing voice quality in the use of the L-3 Maritime-developed SIP solution in progress.

The voice quality testing for the project was done in three parts:

1. Baseline Verification ensures the functionality of individual components test elements, and includes verifying:
 - Equipment/setup
 - Software/setup
 - Test files
2. RTP/Transfer Testing evaluates the performance of the Real-Time Protocol (RTP) to be implemented in the integrated solution. It consists of:
 - Establishing communications links over RTP
 - Transferring selected voice files over the link
 - Evaluating the degradation of files transferred
3. End-to-End Communication Verification tests the VoIP/AS-SIP stack (for call control) and the RTP (for data/voice transfer), and the selected endpoint nodes as an integrated system, and consists of:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- Call completion and traffic handling analysis and evaluation
- Voice file transfer analysis and evaluation

7.2.1 Voice Quality Testing Overview

7.2.1.1 Testing Rationale

Originally, voice quality testing of communications systems consisted almost entirely of subjective measures (for example, those defined in ITU-T P.800).

In subjective tests, human listeners hear and rank the quality of processed voice files according to a certain scale. The most common scale is called MOS (Mean Opinion Score) and is composed of 5 scores of subjective quality:

- 1 – Bad
- 2 – Poor
- 3 – Fair
- 4 – Good
- 5 – Excellent

The MOS score of a given test is the average of all the ranks voted by the different auditors of the various voice files used.

There are significant disadvantages to subjective tests, however, including:

- Only a limited number of variables and/or constants can be controlled in any given test (or set of tests). For example, a simple test consisting of a voice file sent from an audio device to a receiver and played for the listener has as variables (among others) (in general chronological order). Note that all these factors can be controlled for during test data analysis:
 - The quality of the original file being played
 - The quality of the hardware and software used to play the sample file for transmission
 - The medium of transmission and the factors associated with its elements of quality (including distance, interference, any impedance mismatches, etc.)
 - The quality of the hardware and software used to receive and play back the file
- Many listeners are required to produce a statistically stable score



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- As with any test involving subjective human factors, uncontrolled (and largely uncontrollable) variables will affect test outcomes, including (but not limited to) :
 - General listening ability of the person grading the quality of the playback
 - Health of the grading person
 - ‘Priming’ mechanisms which may affect a person’s grading (listening to extremely high-quality or extremely low-quality audio prior to a test will very likely lower or raise (respectively) test scores given
 - Mood of the person listening
 - When repeating the test procedure to the same listening group, the test results may increase significantly by the learning effect--the listeners get familiar with the speech they hear and they understand it better after every listening session. Concentration problems, on the other hand, may decrease the results.

Note that virtually none of these factors can be controlled for during test data analysis. To eliminate the uncontrollable variables associated with subjective testing, more automated and objective methodologies have been developed.

Generally speaking, the methodologies consist of using stored audio files of known recording quality and parameters to be played and transmitted to a receiving device of known quality and parameters. The comparison of data points associated with the original file played and transmitted (the ‘reference’ file) and the file received (the ‘degraded’ file) can computationally yield measures of quality for the transmitting, transporting, and playing mechanisms of the transfer. (See the section ‘Voice Quality Metrics’ for details.)

7.2.1.2 Benefits of Testing

The benefits of performing voice quality testing can be expressed on two levels:

1. The testing of any given platform can determine its suitability for a given application and whether or not the platform meets required specifications. (For examples of specifications, see Department of Defense Unified Capabilities Requirements 2008 (UCR 2008), section 5.2.12.8.2.1: VoIP System Voice Quality.)
2. Testing voice quality with controlled configurations and parameters allows for comparisons of variable elements.

For example, once a set of baseline data is established for a given codec-transfer-codec combination, the underlying network topology and/or protocols can be changed and the results compared.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

A major factor in the selection of tools, methods, and procedures for the testing was to realize the following benefits: meeting the specification requirement documentation and using controlled and automated elements for performing, recording, and analyzing the tests; and modularizing and isolating the elements of tests so that elements (hardware, transfer media, protocols, etc.) can be legitimately introduced, changed, and compared.

7.2.1.3 Summary of Testing

Applicable hardware will be connected in accordance with GL Communications Voice Quality Testing (VQT) User Guide, Updated June 2010 (Reference 3)*.

Applicable software is configured in accordance with guidelines for test types:

- Loopback Testing and Verification (Baseline Verification) (Appendix E),
- Real-Time Protocol (RTP) Testing (Appendix F),
- End-to-End Communication Verification (Appendix D)

They are extractions from the GL Communications manuals with enhancements for the type of environment that will be tested.

Tests for each Stage consist of three iterations ('runs') of each:

- 1 encoded voice file transferred 20 times
- 1 encoded voice file transferred 250 times
- 4 different encoded voice files transferred 10 times
- 4 encoded different voice files transferred 250 times
- 6 different encoded voice files transferred 10 times
- 6 different encoded voice files transferred 250 times

All transfers are bidirectional, with each DUT transmitting and receiving.

The results of each test run will be imported to a spreadsheet, which will be combined and analyzed with the results of other test runs to provide data for the following calculations and analyses (among others):

- Overall performance of test setup hardware at the component level (isolatable by controlling for hardware setups (see Loopback Testing and Verification (Baseline Verification) in Appendix E)
- Performance and voice transfer quality changes at each phase of integration: baseline (loopback), RTP, and end-to-end
- Performance and voice transfer quality changes due to external considerations (network traffic, endpoint configuration, etc.)

*GL Communications Voice Quality Testing (VQT) User Guide, Updated June 2010



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

7.3 Voice Quality Metrics

7.3.1 One Way Delay

One Way Delay (OWD) is the measure of time between the transmission of a signal or file and its reception by the receiver. This metric can indicate the effects and impact of network latency, equipment lag, et cetera.

7.3.2 Round Trip Delay

Round Trip Delay (RTD) is the measure of time between the transmission of a signal or file to a receiver and its return. This metric can indicate the effects and impact of network latency, equipment lag, et cetera, as well as any wait times associated with the receiver of the file or signal and its retransmission.

7.3.3 Mean Opinion Score

Voice Quality:

1. PAMS_LE
2. PAMS_LQ
3. PSQM
4. PSQM_MOS
5. PSQM+
6. Calculated PSQM+_MOS
7. PESQ_Score
8. PESQ_LQ
9. PESQ_MOS

Following are details for each of the Voice Quality parameters.

7.3.4 PAMS_LE, PAMS_LQ

The Perceptual Analysis/Masurement System (PAMS) was developed by British Telecom, and uses a mathematical model of human hearing. PAMS measurements report two scores: Listening Quality (LQ) and Listening Effort (LE).

These are both on a five-point scale similar to the MOS score.

Note: Research shows that PAMS scores average within a half-point when compared to traditional MOS scores.

Note: In PAMS measurements, errors in input signals are taken into consideration.

7.3.5 PSQM and PSQM+



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The Perceptual Speech Quality Measure (PSQM) was developed by the ITU (International Telecommunications Union) as standard P.861. It is designed to measure the effect of compression on voice quality; it does not take lost packets into account.

PSQM scores are on a scale of 0 to 6.5, with 0 indicating no distortion and 6.5 indicating extreme distortion (thus, the lower the number, the better the voice quality). There is no direct correlation between 'traditional' MOS scores and PSQM scores.

PSQM+ was developed to address some of these issues. PSQM+ performs additional post processing on PSQM values, and its results correlate more directly with 'traditional' MOS scores. It is to be noted, however, that PSQM+ still has issues with testing which involves lost packets.

7.3.5.1 Results Provided by PSQM

The VQT performs the PSQM measurement if the algorithm is licensed and the option is selected. The implementation of PSQM is based upon ITU-T Rec. P.861. The algorithm and functionality are described in P. 861 and not repeated here.

The mapping of the PSQM value to Mean Opinion Score (MOS) is described in P.861. PSQM score 0 is equivalent to excellent and 6.5 is very poor on the Listening Quality Scale defined in ITU-T Rec. P.800.

To simplify result analysis, a linear re-scaling is used for calculating Calculated_PSQMPlus_MOS:

$$\text{MOS Listening Quality} = 5 - (4 * \text{PSQM}/6.5)$$

7.3.5.2 PESQ

The Perceptual Evaluation of Speech Quality (PESQ) is presently the most advanced method for voice quality measurement. It was designed to succeed PSQM, and is defined in ITU standard P.862.

PESQ combines the techniques of PSQM+ and PAMS, and takes into account distortion, errors, packet loss, and delay.

PESQ measurement uses a sensory model (the comparison of the original ('reference') signal with the received ('degraded') signal), and its scores are analogous to MOS scores, with the best PESQ score which can be achieved being 4.5.

7.3.5.2.1 Method

The result of comparing the reference and degraded signals is a quality score. This score is analogous to the subjective Mean Opinion Score (MOS) measured using panel tests



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

according to ITU-T P.800. PESQ incorporates many new developments that can distinguish it from earlier models for assessing codecs. These innovations allow PESQ to be used with confidence to assess end-to-end speech quality as well as the effect of such individual elements as codecs.

In addition to the standard PESQ score, the GL VQT also provides the PESQ LQ, LQO (P.862.1) and PESQ WB (P.862.2) score. These revised scores exhibits better correlation to subjective listening quality test scores.

PESQ returns a quality score, known as PESQ score, which conforms to ITU-T P.862. PESQ score lies on a scale from -0.5 to 4.5 , though in most cases it is between 1 and 4.5 . PESQ score correlates with subjective quality scores; however, the PESQ score tends to be optimistic for poor quality speech and pessimistic for good quality speech. Alternative mappings for PESQ score have been developed which exhibit a better correlation to subjective test scores. These are referred to as the PESQ-LQ and PESQ-LQO scores.

7.3.5.3 PESQ-LQ

PESQ-LQ scores are closer to the listening quality subjective opinion scale, which is standard in the industry and is defined in ITU-T P.800. Listening quality scores lie between 1 and 5 . PESQ-LQ score lie between 1.0 and 4.5 , because 4.5 is usually the maximum obtained in subjective tests.

Listening Quality Scale:

5	Excellent
4	Good
3	Fair
2	Poor
1	Bad

The score gives a measure of customers' perception of quality. The highest score, 4.5 , means that no distortion is measured. As the amount of distortion increases, the quality decreases.

7.3.5.4 PESQ-LQO (P.862.1)

The aim of a separate recommendation ITU-T P.862.1 is to provide a single mapping from raw P.862 score to the Listening Quality Objective Mean Opinion Score (LQO-MOS). This latest ITU standard improves on the original PESQ (P.862) by correlating better to subjective test results.

7.3.5.5 PESQ and PSQM Compared



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

(*From ITU-T P.862:*) Real systems may include filtering and variable delay, as well as distortions due to channel errors and low bit-rate codecs. The PSQM method as described in ITU-T P.861 (February 1998), was only recommended for use in assessing speech codecs, and was not able to take proper account of filtering, variable delay, and short localized distortions. PESQ addresses these effects with transfer function equalization, time alignment, and a new algorithm for averaging distortions over time. The validation of PESQ included a number of experiments that specifically tested its performance across combinations of factors such as filtering, variable delay, coding distortions and channel errors. It is recommended that PESQ be used for speech quality assessment of 3.1 kHz (narrow-band) handset telephony and narrow-band speech codecs.

7.3.6 MOS

The most familiar voice quality measurement is Mean Opinion Score (MOS). MOS is expressed on a five-point scale, with 5 being the best and 1 the worst. Table 7-1 displays further detail about MOS ratings.

Table 7-1 MOS-1

Rating	Speech Quality	Distortion Level
5	Excellent	Imperceptible
4	Good	Just perceptible, not annoying
3	Fair	Perceptible, slightly annoying
2	Poor	Annoying, but not objectionable
1	Unacceptable	Very annoying, objectionable

A MOS score of 4.0 is considered "toll quality," the quality associated with traditional PSTN service. Originally, MOS scores were purely subjective, based on individuals listening to voice samples and grading them. However, because external factors (including background noise at both ends of the communication path and even the individual's mood) could greatly influence scoring, more scientific ways of quantifying voice quality have been developed.

Note: Historical research indicates that a 3 per cent packet loss results in a MOS score decrease of 0.5 (out of 5)³.

7.4 Phases of Testing

The overall strategy for testing voice and signal quality for the AS-SIP/VoIP implementation is:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- Verify the equipment and software to be used for the testing (Stage I: Baseline Verification)
- Test the functionality and signal fidelity of the Real-Time Protocol (RTP) for transferring voice files (Stage II: RTP/Transfer Testing)
- Test the functionality and signal fidelity of voice files when transmitted from one phone set endpoint to another (Stage III: End-to-End Communication Verification)

The purpose of using this strategy is to verify the functionality and suitability of a given set of testing elements before layers of complexity are added. Thus:

- Stage I verifies the components of the testing system itself;
- Stage II tests the RTP using the testing system; and
- Stage III tests the entire implementation setup using the hardware and software verified by Phases I and II.

Further detail for the three test phases follows.

7.4.1 Stage I: Baseline Verification

Baseline Verification confirms the functionality of the elements of the testing platform which will be used in later tests, which will be executed in greater levels of complexity.

This Stage of testing ensures the functionality of individual components and test elements, and includes verifying:

- Hardware/Setup

The testing system hardware consists of:

- Two equivalent Dell Latitude D630 laptops as processing nodes
- Two GL Communications Universal Telephony Agents (UTAs)
- Connectors and cabling necessary for interconnection

The components will be tested in all processing node/UTA combinations:

- PC1-to-UTA155155 Loopback Transfer
- PC1-to-UTA155156 Loopback Transfer
- PC2-to-UTA155155 Loopback Transfer
- PC2-to-UTA155156 Loopback Transfer
- Software/Setup



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The testing and analysis system software consists of:

- Windows XP SP3 (operating system for processing nodes)
 - GL Communications VQuad (file transfer and traffic generation software)
 - GL Communications VQT (Voice Quality Testing and analysis software)
 - Wireshark and other utilities necessary for support and analysis
 - Microsoft Excel and Word for data analysis and presentation
- Test Files

The files used for the initial three testing phases will consist of voice files provided by GL Communications for use with VQuad/VQT software.

For detailed information on setting up and configuring for Baseline Verification Testing, see Appendix E, Loopback Testing and Verification (Baseline Verification).

7.4.2 Stage II: RTP/Transfer Testing

RTP/Transfer Testing evaluates the performance of the Real-Time Protocol (RTP) to be implemented in the integrated solution. It consists of:

- Establishing communications links over RTP
- Transferring selected voice files over the link
- Evaluating the degradation of files transferred

For detailed information on setting up and configuring for Baseline Verification Testing, see Appendix F, Real-Time Protocol (RTP) Testing.

7.4.3 Stage III: End-to-End Communication Verification

End-to-End Communication Verification tests the VoIP/AS-SIP stack (for call control), the RTP (for data/voice transfer), and the selected endpoint nodes as an integrated system, and consists of:

- Call completion and traffic handling analysis and evaluation
- Voice file transfer and analysis and evaluation

7.4.4 Test Setup

7.4.4.1 Stage I: Baseline Verification (Loopback Testing)



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

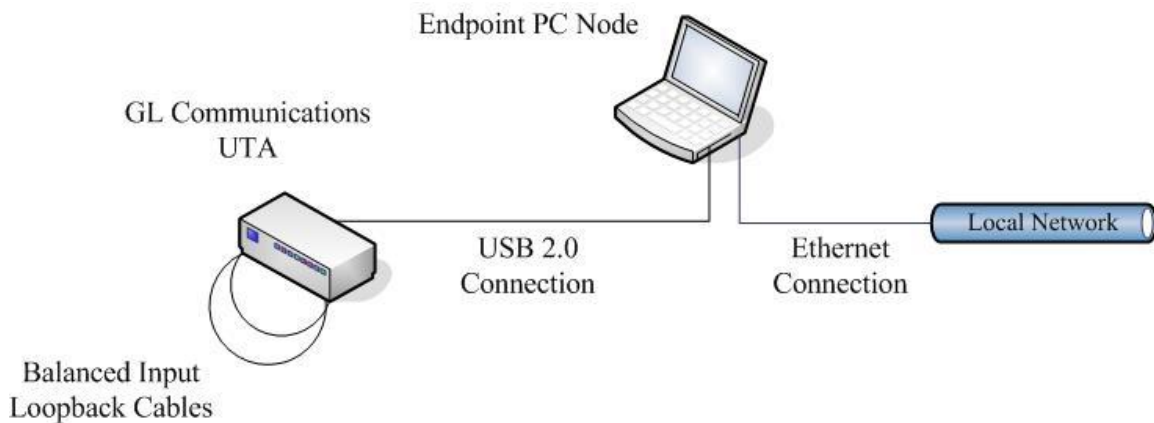


Figure 7-1 Loopback Transfer Test Setup

7.4.4.2 Hardware Setup

The testing system hardware consists of:

- Two equivalent Dell Latitude D630 laptops as processing nodes
- Two GL Communications Universal Telephony Agents (UTAs)
- Connectors and cabling necessary for interconnection

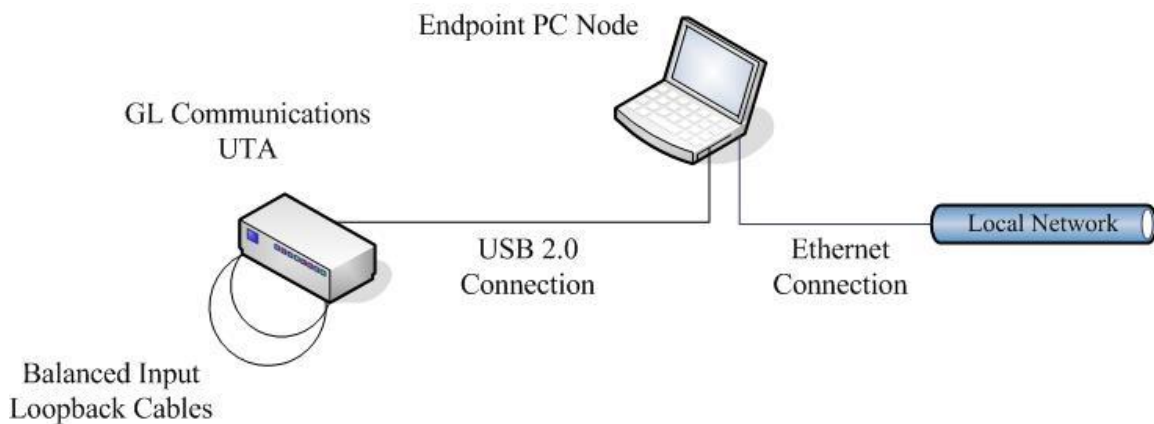


Figure 7-2 Loopback Setup

Initial tests will consist of a single PC and a single UTA. For verification purposes, tests will be run with all PC/UTA combinations (see 'Test Results' for test tabulation summary).

7.4.4.3 Single-PC/Single-UTA Loopback Test Setup



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The configuration used for Loopback testing is a one-node system on a single PC with a Dual UTA (Audio IN to Audio OUT).

1. On the Dual UTA, cross-connect using 3.5mm mono audio cables:
 - a. Audio IN of Side #1 to the Audio OUT of Side #2
 - b. Audio OUT of Side #1 to the Audio IN of Side #2
2. Ensure the UTA is connected to a PC running VQuad™ via a USB 2.0 connection.

7.4.4.4 Data Path

Selected voice files will be transferred from the Endpoint PC Node to the UTA, which will perform an internal loopback transfer using the Balanced inputs and outputs. The transferred files will then be sent to the Endpoint PC Node and stored in the local VQT_Degraded directory (see paragraph 7.4.5, Software Setup, and 7.4.5.1 VQuad Configuration.)

7.4.5 Software Setup

The testing and analysis system software consists of:

- Windows XP SP3 (operating system for processing nodes)
- GL Communications VQuad (file transfer and traffic generation software)
- GL Communications VQT (Voice Quality Testing and analysis software)
- Wireshark and other utilities necessary for support and analysis
- Microsoft Excel and Word for data analysis and presentation

7.4.5.1 VQuad Configuration

To configure GL Communications VQuad for Loopback Testing:

1. Start VQT by double-clicking the GL VQuad icon:

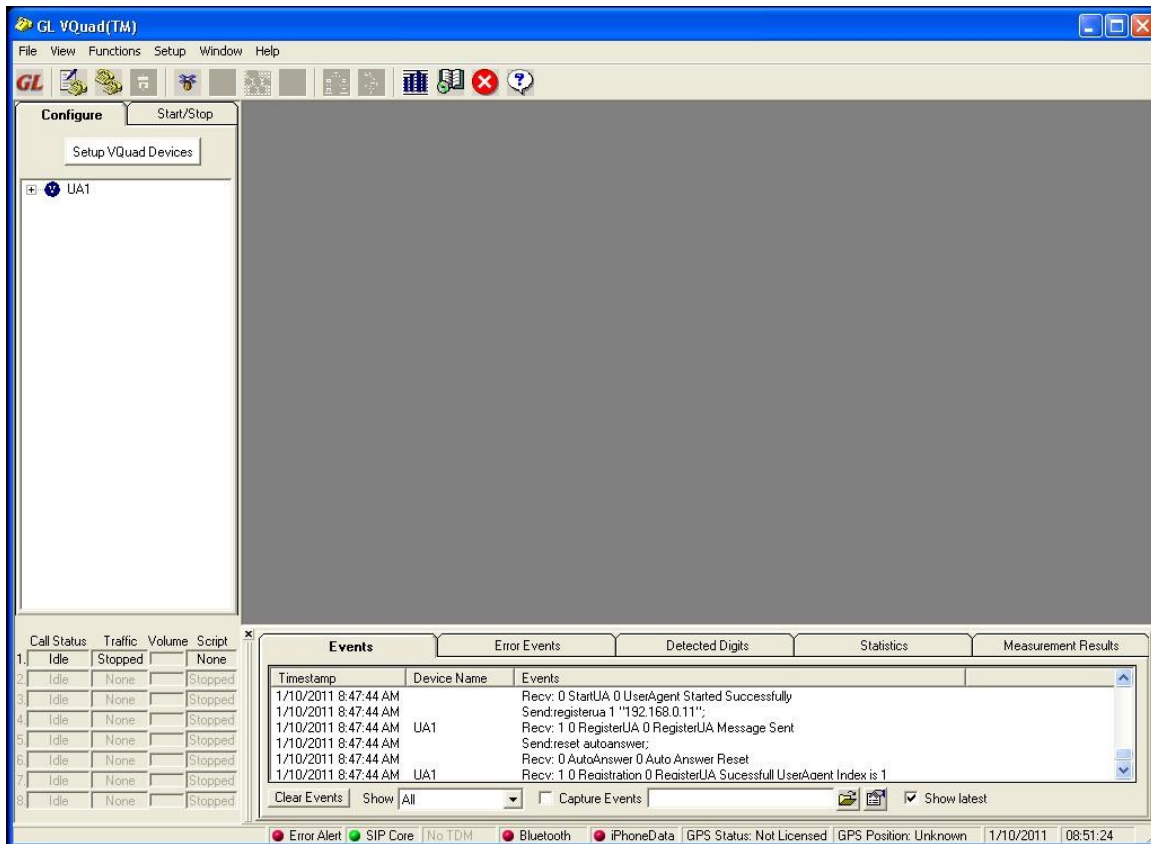


2. The VQuad Main Screen appears.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



3. Click the **Configure** tab:



A custom device configuration has been created for this test implementation. To access and use the configuration, click the **Setup VQuad Devices** button:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



4. Click the **Device Configuration – Load** button:



5. Select the configuration **PCx_Loopback_20x1**.

The naming scheme for device configurations is as follows:

For the configuration **PCx_Loopback_20x1**,

PCx_ indicates the PC testing node for which the configuration is designed:

- PC1 indicates Endpoint PC Node 1
- PC2 indicates Endpoint PC Node 2
- PCx indicates the configuration can be used for either Endpoint PC Node

Loopback_ indicates the type of test:

- Loopback indicates the test is single-node, endpoint-to-same-endpoint
- Transfer indicates the test is multi-node (two or more nodes), transfer endpoints different
- End To End indicates that all devices in a VoIP/AS-SIP communication path will be tested

20x1 indicates the number of transfers and the number of files:

- The first number (in this case, 20) is the total number of transfers to be performed



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- The second number (in this case, 1) is the number of files which will be transferred

Thus, the configuration name **PC2_Transfer_100x4** would indicate:

- The configuration is designed for PC2
- It is a multi-node transfer configuration
- 100 transfers of four files will be performed

6. Click the **Exit** button to save the configuration selection and exit.

7.4.5.2 Modify or Create a VQuad Device Configuration for Loopback Testing

If the required configuration is not available, or another configuration is to be created, use the following procedure:

1. Click the **Device Configure** tab:



2. Click the **Setup VQuad Devices** button:





Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

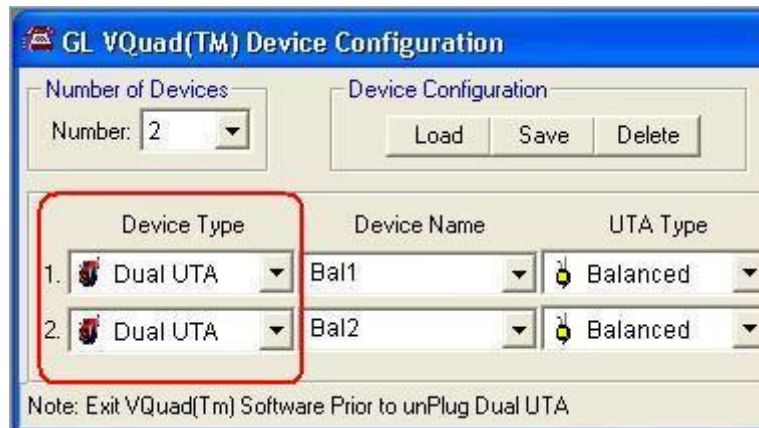
Maritime Systems

For loopback testing, two devices will be set up as ‘Balanced Input’.

3. In the GL VQuad™ Device Configuration window, click the **Number of Devices** down arrow and select **2**.



4. Click the **Device Type** down arrow and select **Dual UTA** as the device type.



5. Enter a name in the **Device Name** field. For illustration purposes, the Device Names entered will be **Bal1** and **Bal2**:





Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

6. Select **UTA Type** as **Balanced I/O** for both ports. (Dual UTA port numbers are automatically displayed with the loopback connection):

The screenshot shows the 'GL VQuad(TM) Device Configuration' window. At the top, there's a 'Number of Devices' section with a dropdown set to '2'. To the right is a 'Device Configuration' section with 'Load', 'Save', and 'Delete' buttons. Below this is a table with three columns: 'Device Type', 'Device Name', and 'UTA Type'. The table contains two rows: 1. 'Dual UTA' (with a device icon), 'Bal1', and 'Balanced' (with a device icon). 2. 'Dual UTA' (with a device icon), 'Bal2', and 'Balanced' (with a device icon). A red rectangle highlights the 'UTA Type' column for both rows. At the bottom, a note reads: 'Note: Exit VQuad(TM) Software Prior to unPlug Dual UTA'.

	Device Type	Device Name	UTA Type
1.	Dual UTA	Bal1	Balanced
2.	Dual UTA	Bal2	Balanced

7. Save the device configuration with a descriptive name using the **Save** button. For illustration purposes, the configuration will be saved as **Balanced_Input_1**:

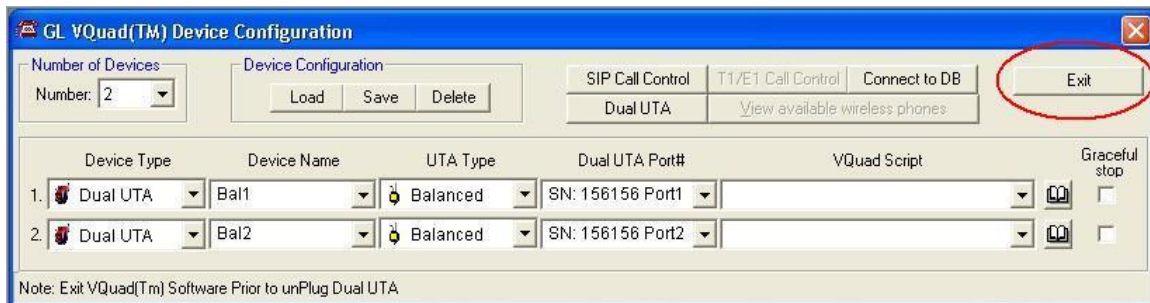
This screenshot is identical to the previous one, but with a red circle highlighting the 'Save' button in the 'Device Configuration' section at the top right of the window.

8. Click the **Exit** to save the configuration and exit **Device Configuration**:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



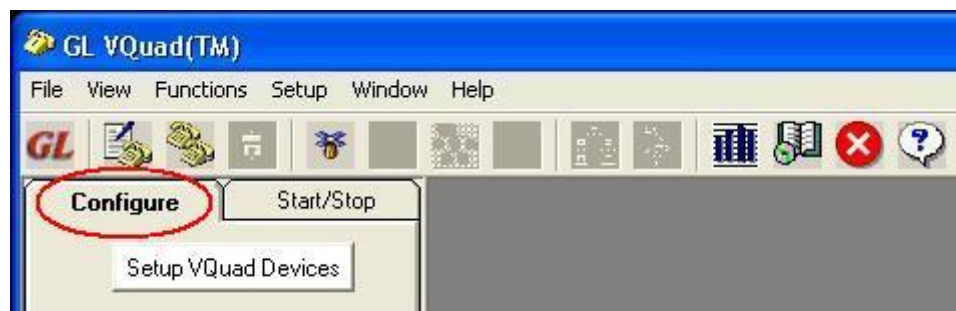
VQuad devices are now configured to perform loopback transfers using the two balanced inputs on the attached UTA.

7.4.5.3 Auto-Traffic Configuration

Auto-Traffic Configuration determines the method of transfer (master/slave identification), the type of file transferred, and the locations of files transmitted and received.

Custom configurations have been created for this test implementation. To access and use the configurations, use the following procedure:

In the VQuad Main Window, click the **Configure** tab:



7.4.5.3.1 Configure Device 1

1. Double-click **Bal1** (or any other name configured to Device 1 in the previous procedure) in the Devices panel on the left of the VQuad Main Window. Alternately, click the + sign to the left of the named device:



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

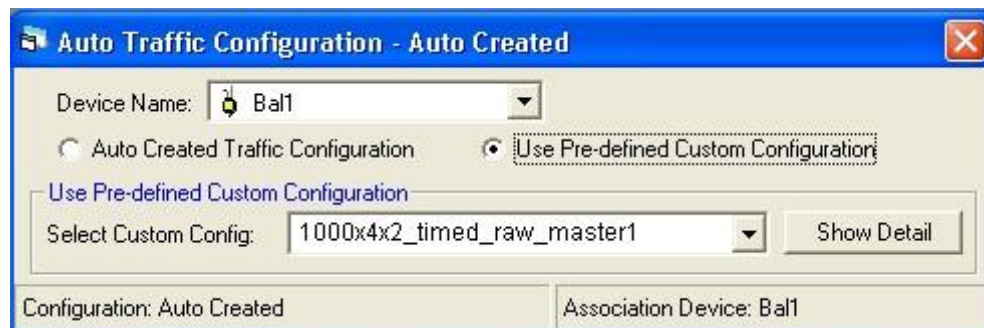


The Bal1 detail categories will be listed:



2. Double-click the **Auto-Traffic** entry.

The Auto-Traffic Setup window will appear:

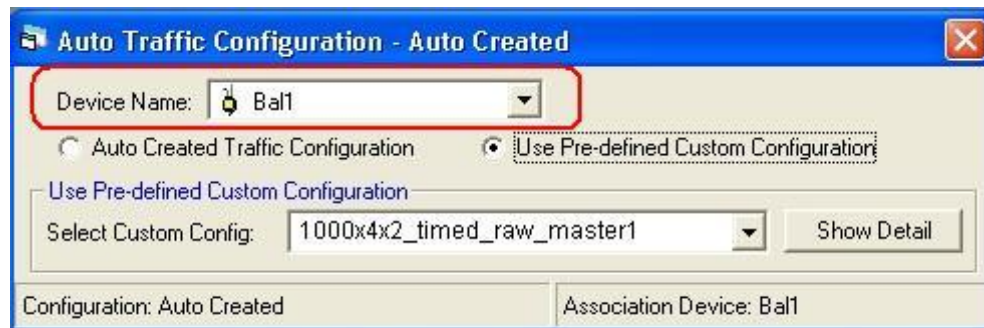


3. Select **Bal1** as the **Device Name**:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



- Click the Use Pre-defined Custom Configuration radio button:



- Click the down arrow to the right of the Configuration Name field to access the list of available configurations.
- Select the configuration **Loopback_20x1_Master** for Device 1.

The naming scheme for Auto-Traffic configurations is as follows:

For the configuration **Loopback_20x1_Master**,

Loopback indicates the type of test:

- Loopback** indicates the test is single-node, endpoint-to-same-endpoint
- Transfer** indicates the test is multi-node (two or more nodes), transfer endpoints different
- EndToEnd** indicates that all devices in a VoIP/AS-SIP communication path will be tested

20x1 indicates the number of transfers and the number of files:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- The first number (in this case, **20**) is the total number of transfers to be performed
- The second number (in this case, **1**) is the number of files which will be transferred

Master indicates the selected device's role in the transfers:

- **Master:** In this mode, the VQuad™ will be in master configuration and transmits the reference files at a regular time interval (set by the operator) regardless of whether or not it receives any incoming degraded files from the other device.
- **Slave:** This mode transmits a VQuad™ reference file only in response to receiving an incoming degraded file.

Thus, the configuration name **PC2_Transfer_100x4** would indicate:

- a. The configuration is designed for PC2
- b. It is a multi-node transfer
- c. 100 transfers of four files will be performed

After selecting the configuration **Loopback_20x1_Master**, ensure that the following parameters are correct for the relevant device:

Auto Traffic Configuration - Loopback_20x1_Master

General Tx Provisioning Rx Provisioning

Auto Trigger

Master Slave

☒ Trigger On Time ☐ Trigger On DTMF digits ☐ Trigger On MF digits ☐ Trigger On User Defined Tones

Trigger On Time

Cycle Time (s): 30 Loop Period (s): 120

Stop Iterations: 20

File Length (s): 8 Tx/Rx Offset (s): 6

Send Delay (ms): 1500

Load Configuration Save As Configuration

Configuration: Loopback_20x1_Master Association Device: Bal1

Figure 7-3 Loopback_20x1_Master Auto-Trigger Tab



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

The screenshot shows the 'General' tab of the 'Auto Traffic Configuration - Loopback_20x1_Master' window. The 'File Format' section has three radio buttons: '8000; 8-bit; A-Law', '8000; 8-bit; Mu-Law', and '8000; 16-bit; PCM', with the last one selected. The 'Increment Rx File Name' section has three radio buttons: 'Sequential', 'Time Stamp' (selected), and 'GPS + Time Stamp / ITS + Time Stamp'. The 'Digit Parameters' section has two checkboxes: 'Enable Digit Tx' and 'Enable Multiple Digit Detection', both unchecked. There are four spin boxes: 'On Time (ms)' set to 200, 'Off Time (ms)' set to 750, 'High Power (-dB)' set to 10, and 'Low Power (-dB)' set to 10. At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons. The status bar shows 'Configuration: Loopback_20x1_Master' and 'Association Device: Bal1'.

Figure 7-4 Loopback_20x1_Master General Tab

The screenshot shows the 'Tx Provisioning' tab of the 'Auto Traffic Configuration - Loopback_20x1_Master' window. The 'Timing' section on the left has a list with values 0, 30, 60, and 90. The 'Reference File Path' section has a text box containing 'C:\WQT_Reference\VQuad_Auto\Raw\em1_1.pcm' and a folder icon button. Below this are seven empty text boxes, each with a folder icon button to its right. The 'Cycle Time(ms)' section has a spin box set to 2000. The 'Identifier Digit' section has two radio buttons: 'In File' (selected) and 'Generate'. The 'Power (dB)' section has a spin box set to 0. The 'Duration (ms)' section has a spin box set to 0. At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons. The status bar shows 'Configuration: Loopback_20x1_Master' and 'Association Device: Bal1'.

Figure 7-5 Loopback_20x1_Master Tx Provisioning Tab



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

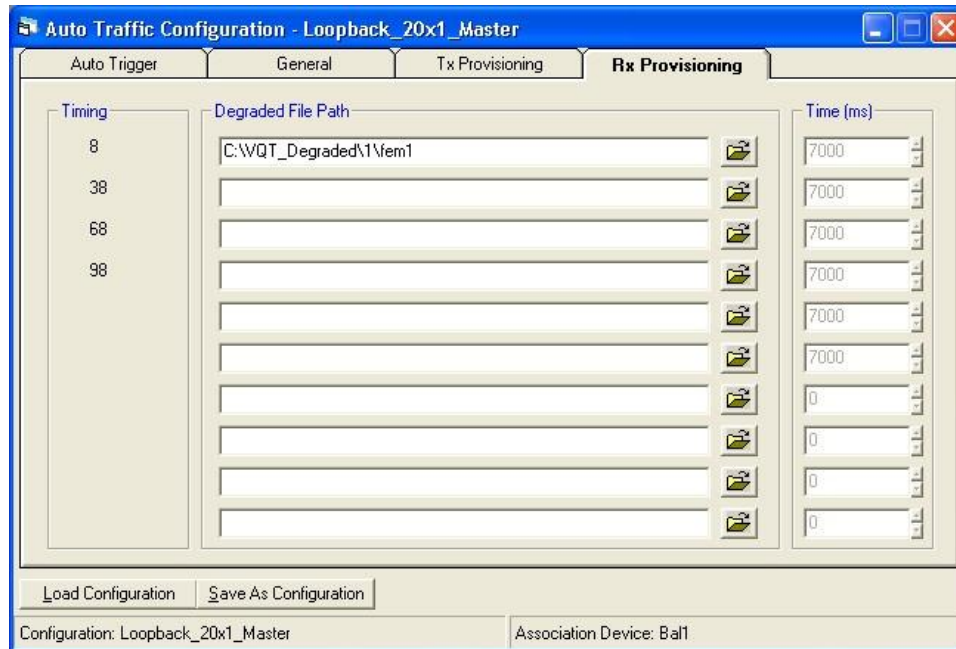


Figure 7-6 Loopback_20x1_Master Rx Provisioning Tab

7.4.5.3.2 Configure Device 2:

1. Double-click **Bal2** (or any other name configured to Device 1 in the previous procedure) in the Devices panel on the left of the VQuad Main Window. Alternately, click the + sign to the left of the named device:



The Bal2 detail categories will be listed:



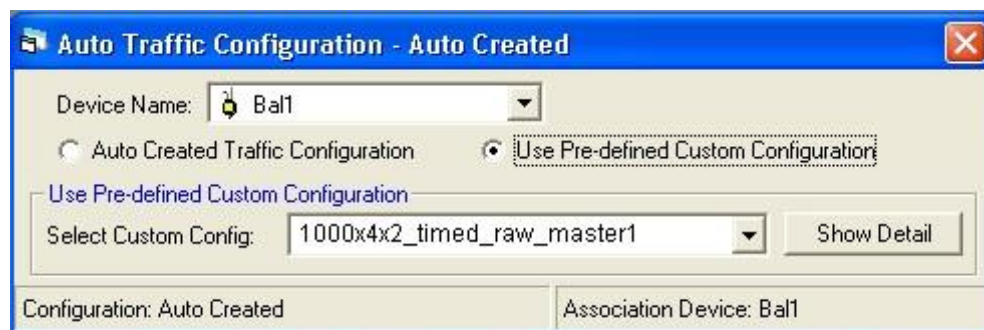
Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

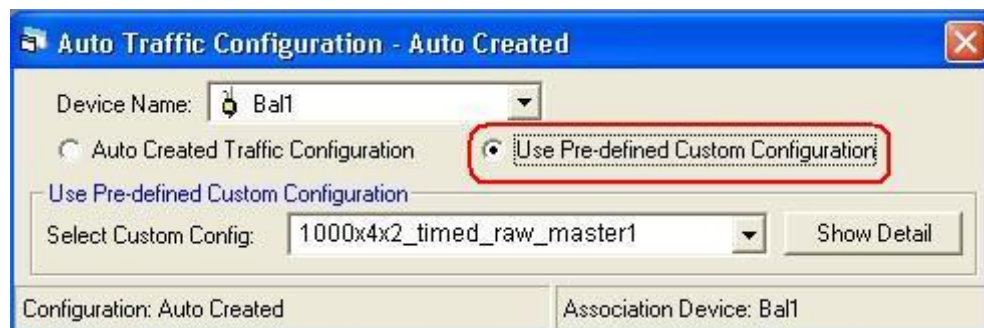


Double-click the **Auto-Traffic** entry.

The Auto-Traffic Setup window will appear:



2. Select **Bal2** as the **Device Name**:
3. Click the Use Pre-defined Custom Configuration radio button:





Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

4. Click the down arrow to the right of the Configuration Name field to access the list of available configurations.
5. Select the configuration **Loopback_20x1_Slave** for Device 2.

After selecting the configuration **Loopback_20x1_Slave** for Device 2, ensure that the following parameters are correct for the relevant device:

The screenshot shows a software window titled "Auto Traffic Configuration - Loopback_20x1_Slave". It has four tabs: "Auto Trigger" (selected), "General", "Tx Provisioning", and "Rx Provisioning". In the "Auto Trigger" tab, there are two sub-tabs: "Master" and "Slave", with "Slave" selected. Below these are four radio buttons: "Trigger On Time" (selected), "Trigger On DTMF digits", "Trigger On MF digits", and "Trigger On User Defined Tones". Under the "Trigger On Time" section, there are several input fields: "Cycle Time (s)" set to 30, "Loop Period (s)" set to 120, "Stop Iterations" set to 20, "File Length (s)" set to 8, "Tx/Rx Offset (s)" set to 6, and "Send Delay (ms)" set to 1500. At the bottom of the window, there are two buttons: "Load Configuration" and "Save As Configuration". The status bar at the very bottom shows "Configuration: Loopback_20x1_Slave" and "Association Device: Bal1".

Figure 7-7 Loopback_20x1_Slave Auto-Trigger Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'General' tab of the 'Auto Traffic Configuration - Loopback_20x1_Slave' window. The 'File Format' section has three radio buttons: '8000; 8-bit; A-Law', '8000; 8-bit; Mu-Law', and '8000; 16-bit; PCM', with the last one selected. The 'Increment Rx File Name' section has three radio buttons: 'Sequential', 'Time Stamp', and 'GPS + Time Stamp / ITS + Time Stamp', with 'Time Stamp' selected. The 'Digit Parameters' section has two checkboxes: 'Enable Digit Tx' and 'Enable Multiple Digit Detection', both unchecked. To the right of these checkboxes are four spin boxes: 'On Time (ms)' set to 200, 'High Power (-dB)' set to 10, 'Off Time (ms)' set to 750, and 'Low Power (-dB)' set to 10. At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons. The status bar shows 'Configuration: Loopback_20x1_Slave' and 'Association Device: Bal1'.

Figure 7-8 Loopback_20x1_Slave General Tab

The screenshot shows the 'Tx Provisioning' tab of the 'Auto Traffic Configuration - Loopback_20x1_Slave' window. The 'Timing' section on the left has a list with values 0, 30, 60, and 90. The 'Reference File Path' section in the center has a text box containing 'C:\WQT_Reference\WQuad_Auto\Raw\mem1_1.pcm' and a browse button. Below this are seven more empty text boxes, each with a browse button. The 'Hold Off Time(ms)' section on the right has a spin box set to 2000. The 'Identifier Digit' section has two radio buttons: 'In File' and 'Generate', with 'In File' selected. Below this are 'Power (dB)' and 'Duration (ms)' spin boxes, both set to 0. At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons. The status bar shows 'Configuration: Loopback_20x1_Slave' and 'Association Device: Bal1'.

Figure 7-9 Loopback_20x1_Slave Tx Provisioning Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

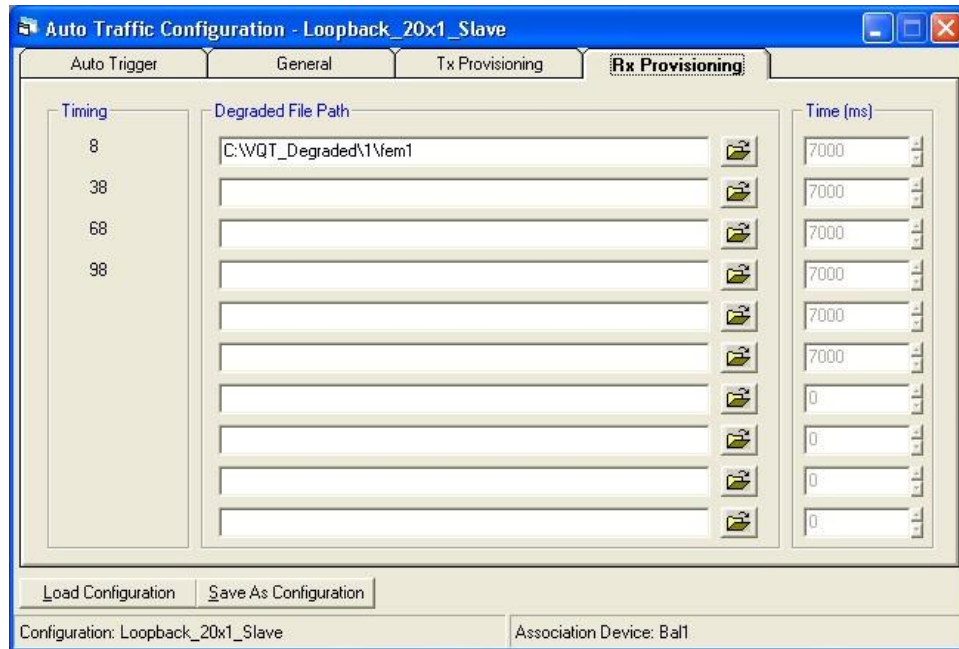


Figure 7-10 Loopback_20x1_Slave Rx Provisioning Tab

7.4.5.3.3 Modify or Create a VQuad Auto Traffic Configuration

If the required configuration is not available, use the following procedure to modify it or create a new one:

Configure the Auto Trigger, General, Tx Provisioning, and Rx Provisioning tabs in accordance with applicable requirements.

7.4.5.4 Configuring Tab Fields

Guidelines for configuring the fields of the tabs follow:

7.4.5.4.1 Auto Trigger Tab Fields

- **Cycle Time:** This is the time allotted for the VQuad™ master and slave operations to take place. This time should be adequately long when bidirectional operations are performed.
- **Loop Period:** This is the total length of a timed cycle. (The default profile uses six files in a cycle.)
- **Stop Iteration:** Specifies the number of iterations (cycles multiplied by number of lines configured) that the VQuad™ will execute before stopping the test.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- **File Length:** This indicates the recording duration and is required to set up the VQuad™ time triggering flow. This includes the time to receive an entire file length plus the send delay time.
- **Tx/Rx Offset:** This is the ‘wait period’ after completion of a Tx action, but before the return of Tx action for bi-directional file transfer.
- **Send Delay:** This is the time after which the VQuad™ receiving end starts recording. This is a buffer time which helps counter the clock variations of two independent PCs.

7.4.5.4.2 General Tab Fields

- **File Format:** This setting selects the format of the file that the VQuad™ receiver will be saved to after the receiver completes recording of an incoming degraded file. The recorded degraded file type must match the type of VQuad reference file being sent, or the VQT Analysis program will yield very bad VQT scores due to mismatched file types. VQuad reference files are provided in linear PCM, A-law, and Mu-law encoding formats. There are usually small differences in VQT scores when using one file type as opposed to another. VQuad™ normally executes using linear PCM Reference files.
- **Note:** When performing transfers using Dual UTAs, the file type must be 16-bit 8000 Hz sampled, Little Endian (Intel) PCM.
- **Increment Rx File Name :** When the VQuad™ completes recording of an incoming degraded voice file, the file several options can be used to suffix the received file name to simplify identifying the degraded files.

The filename prefix for each recorded sample is defined by the filename rules setup in the Rx Provisioning tab.

- **Sequential:** The sequential option simply appends a sequential number to each sample filename in the order as they are received. By default, the numbering starts from zero, but the user can select Sequential File Numbering to start the numbering sequence at any number, or to reset a sequence that was previously started.
- **Time Stamp:** Each incoming/recorded-degraded file has the time stamp (from the PC clock) that the sample was received as the filename suffix. (Note: Ensure that the PC clock is set to the correct time.)



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- Digit Parameters
 - The Digit/Tone Parameters only work in conjunction with the User-defined Tone Triggering method when trigger tones external to the sample voice file are required.
 - If Enable Digit TX is not checked, the VQuad™ program uses MF, DTMF or User-defined tones, attached (prefixed) to the VQT Reference File, to trigger the recording and to identify the VQT Reference File that follows. If Enable Digit TX is checked, the VQuad™ program sends tones according to the table of user-defined tones before sending voice file. In this case, the reference voice files without prefixed tones must be used. (Refer to section Tone/Digit Method for Sending/Recording for details)
 - Enable Multiple Digit Detection is used only in ‘master mode’ with DTMF/MF trigger. It allows multiple digits detection during one session.
 - For example, in normal case (Enable Multiple Digit Detection is not checked), one session consists of a ‘TX file’ and a ‘RX file’. Even if it detects another digit, it will wait till the end of the cycle duration and then proceeds to the next session. However, the Enable Multiple Digit Detection option when checked, will again allow “RX” file event to occur after another digit is detected.
 - Enter On Time (200ms or longer), Off Time (750ms or longer), High Power (-dB) and Low Power (-dB) values.

7.4.5.4.3 Tx Provisioning Tab Fields

- Reference File Path: This field determines the location of the file(s) to be transferred.

Sample files are provided by GL Communications, and are located in local subdirectories of **C:\VQT_Reference**.

7.4.5.4.4 Rx Provisioning Tab Fields

- Degraded File Path: The path on the local PC which is the destination for files transferred in the assigned time slot.

The final characters of the **Degraded File Path** entry will become the initial characters of the files placed in the Degraded File Path directory. These characters will be suffixed with the **Sequential** or **Time Stamp** data (configured in the **General** tab).

7.4.6 VQT Setup

To configure GL Communications VQT for Auto Measurement Setup and Execution:



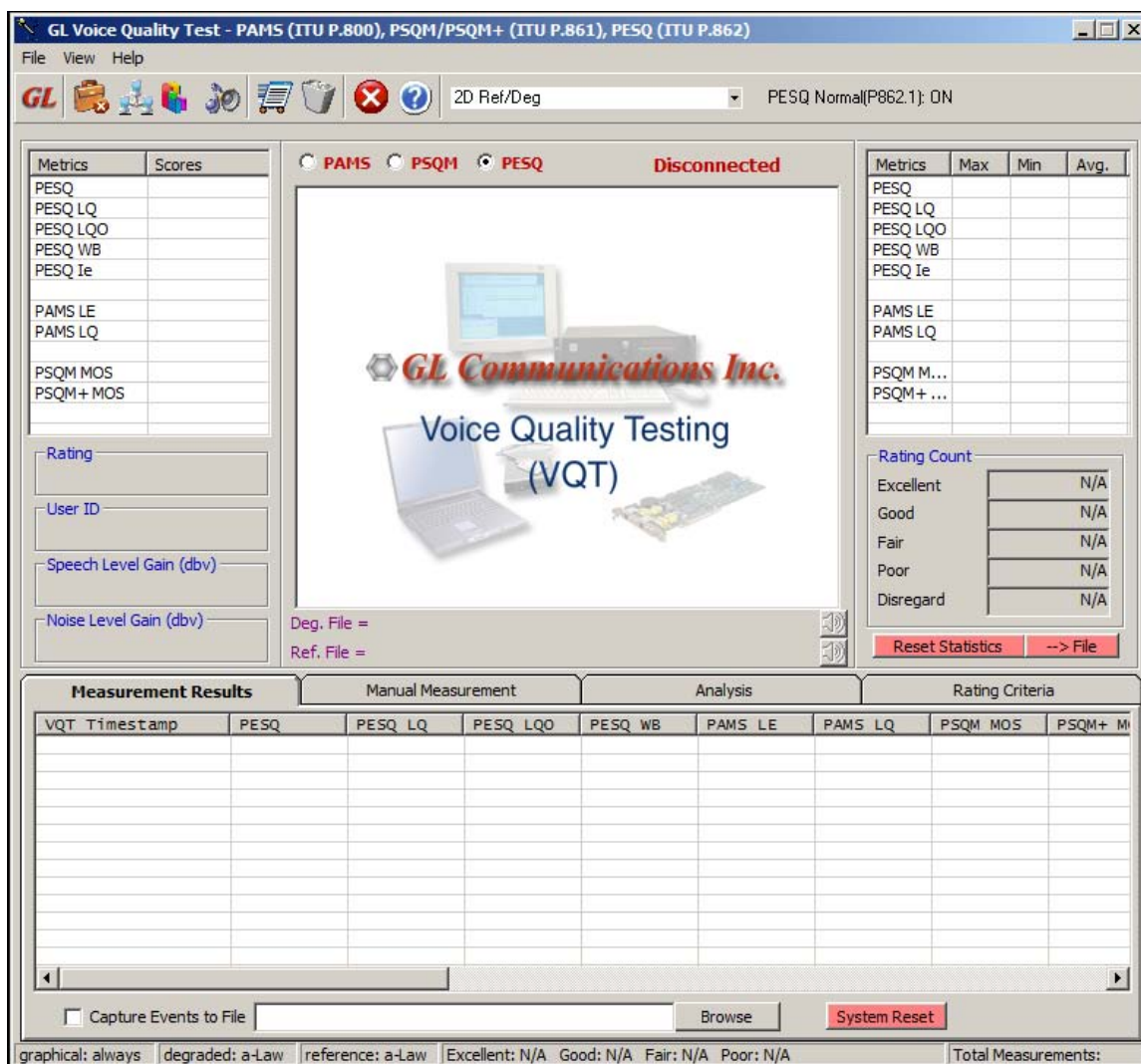
Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

1. Start VQT by double-clicking the GL Voice Quality Testing icon:



2. The VQT Main Screen appears:

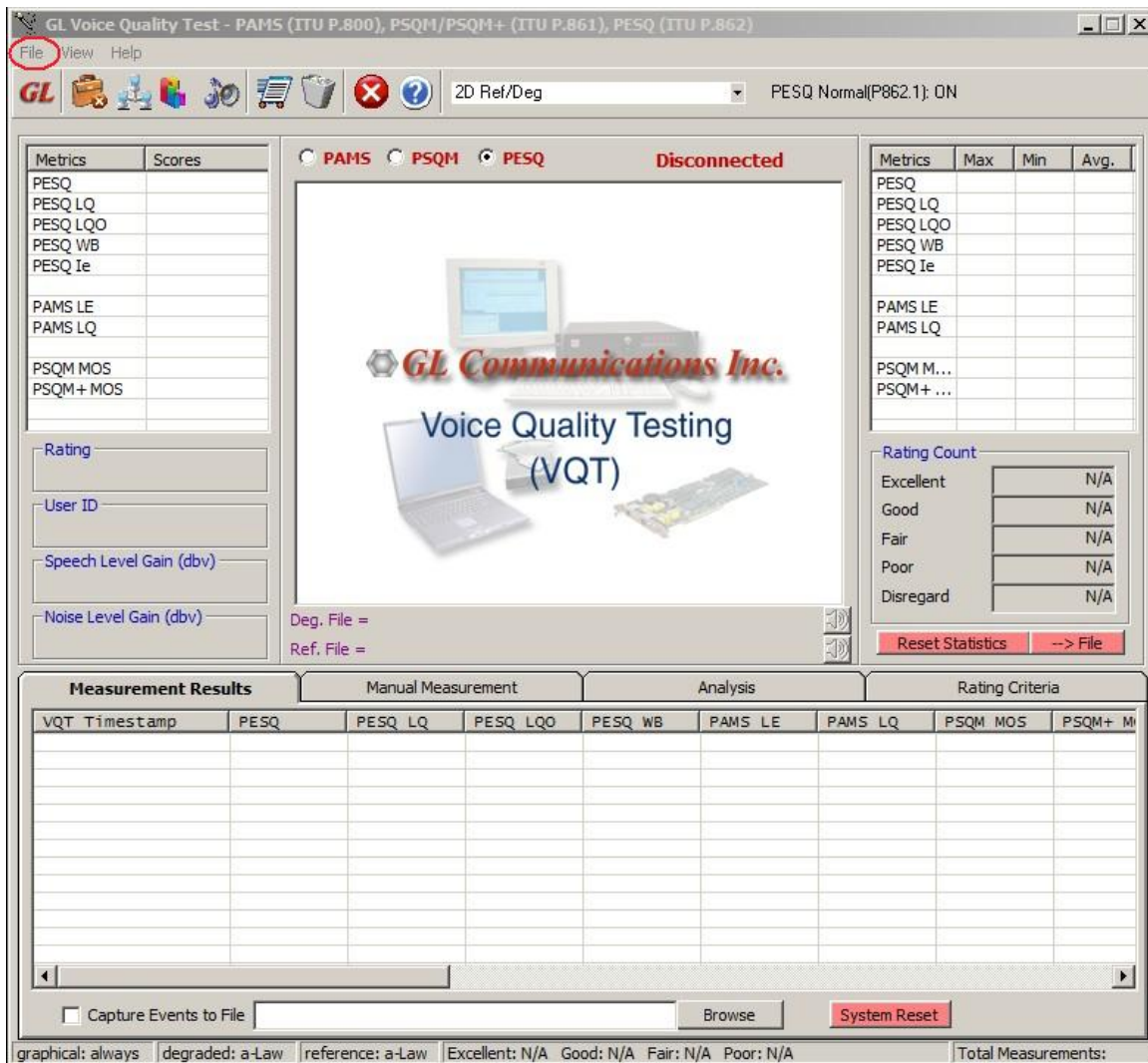


3. From the VQT main screen, select **File -> Auto Measurement**:

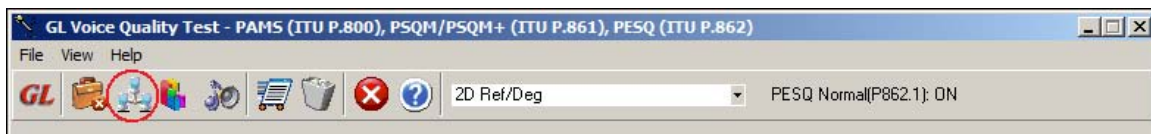


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



Alternatively, the **Auto-Measurement** icon can be clicked:



4. The **VQT Auto-Measurement** screen appears:

[illegible]

The screenshot shows the "VQT Auto-Measurement" application window. At the top is a title bar with the text "VQT Auto-Measurement" and a close button. Below the title bar is a menu bar with "File" and "Help". The main area contains a table with seven columns: "Degraded Directory", "Reference File", "Type", "Option", "Inventory", "User ID", and "Counts". The table has multiple empty rows. At the bottom of the window is a toolbar with several buttons: "Add", "Modify", "Delete", "Delete All", "Start", "Stop", "Start All", and "Stop All". The "Add" button is highlighted with a red circle.

170



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

7. Enter the **Degraded Directory** in the space provided. This is the directory containing the files transferred in the VQuad Loopback Test, and is configured in VQuad Auto Traffic Configuration. (See **VQuad Auto Traffic Configuration – Rx Provisioning Tab.**)

For illustration purposes, **C:\VQT_Degraded\1** will be entered.

Note: Ensure that **C:\VQT_Degraded\1** (or whatever directory is specified as the **Degraded Directory**) exists on the hard drive. Clicking the **Open Folder** icon will open a Windows selection dialogue which will allow selection from existing directories:

8. Enter the **Reference File** in the space provided. This is the original file which was transferred according the VQuad Loopback Test, and is configured in



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

VQuad Auto Traffic Configuration. (See **VQuad Auto Traffic Configuration – Tx Provisioning Tab.**)

For illustration purposes, C:\VQT_Reference\VQuad_Auto\fem1.pcm will be selected.

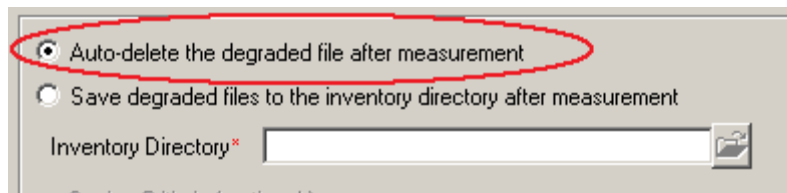
9. Select the **Save degraded files to an inventory directory after measurement** radio button.
10. Enter the **Inventory Directory** in the space provided. This is the directory to which files will be transferred after testing. Thus, the file path is:

Degraded directory (specified in Step a) above)

For illustration purposes C:\VQT_Degraded\Inventory\1\ will be selected.

Note: Ensure that C:\VQT_Degraded\Inventory\1 (or whatever directory is specified as the **Degraded Directory**) exists on the hard drive. Clicking the Open Folder icon will open a Windows selection dialogue which will allow selection from existing directories.

Alternately, the **Auto-delete the degraded file after measurement** radio button can be selected:



If this button is selected, the transferred files will be deleted from the Degraded directory after testing. This option may be preferred if the test parameters are known correct, and the test measurements will not be repeated.

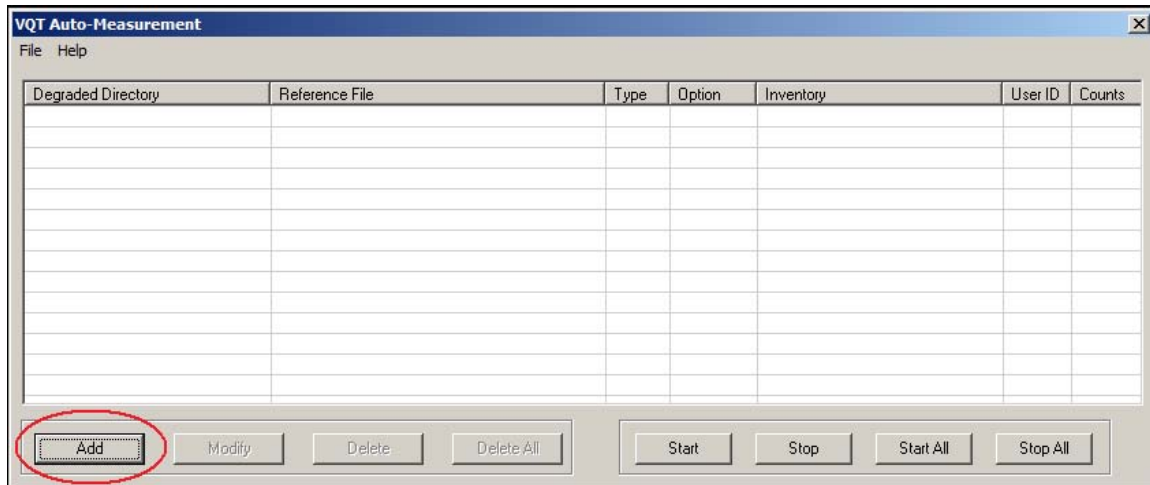
However, for purposes of data retention, or if running any given tests again is necessary or possible, the use of the **Save degraded files to an inventory directory after measurement** is preferable.

11. Within the **User ID** field, enter a User ID for tracking purposes. This field is not verified or cross-checked; it is stored as part of the data produced by the test, and can be useful for granular data analysis in a multi-testing environment.
12. Select the **Add** button to add the session:

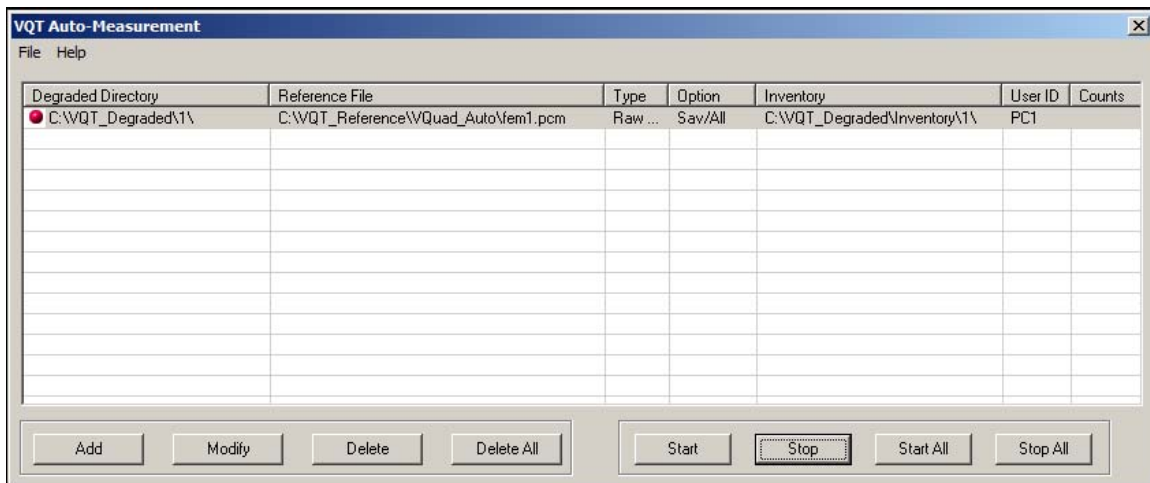


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



13. The **VQT Auto-Measurement** window will now indicate the data associated with the test entered:



7.4.6.1 Checking VQT Auto-Measurement Setup

To verify that the VQT setup is functional, manually copy a file of the correct format into the specified Degraded directory. If setup is correct, upon test execution VQT will automatically run the measurement on the file and move the file to the Inventory directory.

1. Click the Start or Start All button to begin the test:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



2. If the **Start** button is clicked, only the test highlighted in the list above will be started. If the **Start All** button is clicked, all tests in the list above will be started.

If the specified Degraded directory contains one or more files when VQT testing is started, VQT will immediately begin processing the files. The quality tests will be performed and tabulated, then the files tested will be moved to the specified Inventory directory.

If the specified Degraded directory does not contain files when VQT testing is started, VQT will wait for a file to be placed in the directory. When a file is placed in the directory (usually by VQuad transfer), VQT will automatically perform the measurement and move the file to the specified Inventory directory.

3. When the tests have been verified to run correctly, click the Stop All button to end testing:

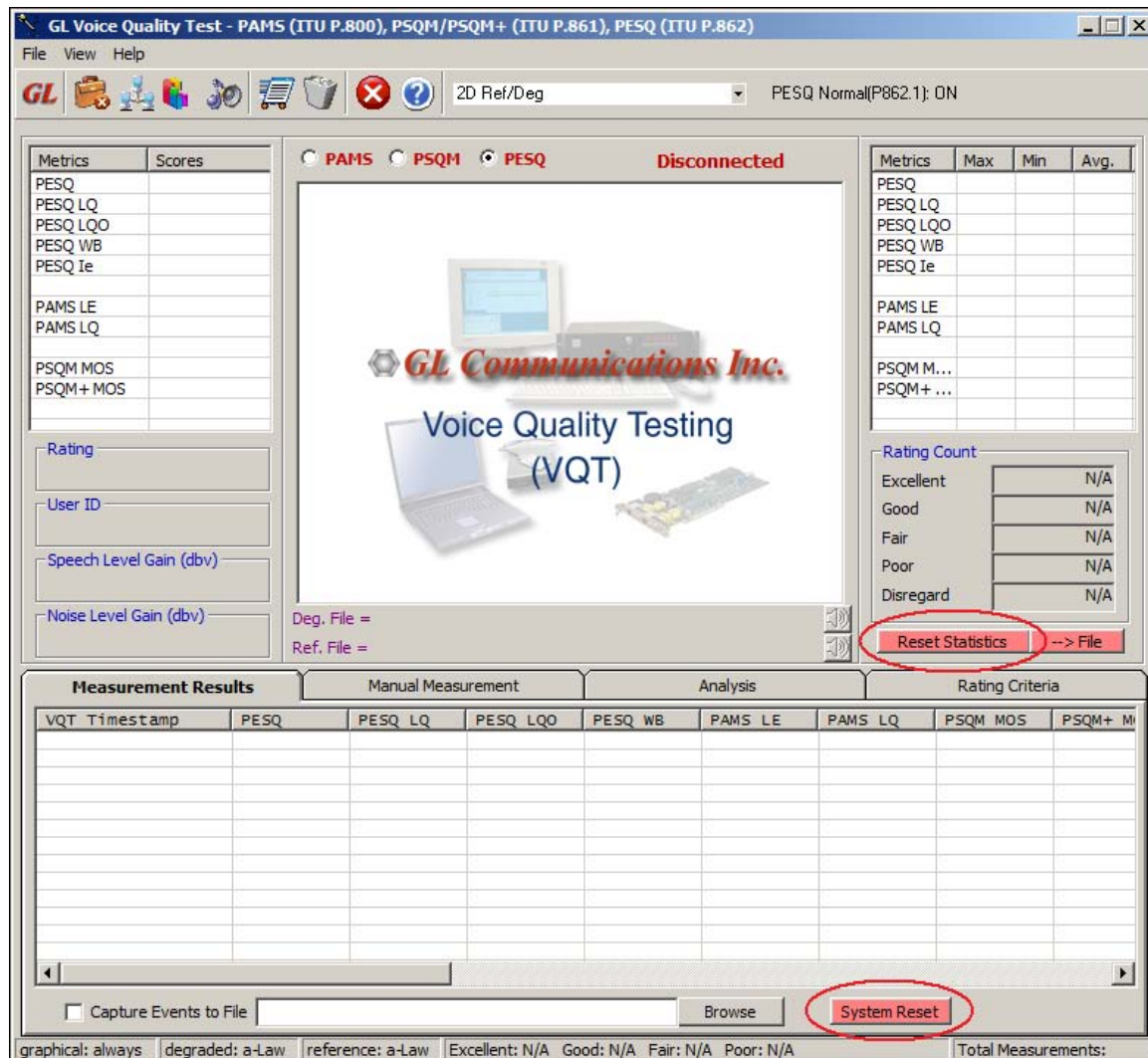


Before starting a test on transferred data, click the **Reset Statistics** and **Reset System** buttons to clear the display and measurement files:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



7.4.6.2 Test Execution

With the above configurations in place, the system is prepared to perform automated file transfer and voice file quality testing.

7.4.6.3 Preparation

1. Ensure that the **Reference** files specified in the **VQuad Auto-Traffic Tx Provisioning** and the **VQT Reference File** sections exist, and are of the correct (and matching) file format(s).
2. Ensure that the **VQT_Degraded** directory/directories specified in the **VQuad Auto-Traffic Tx Provisioning** section exist and are empty.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

3. Ensure that the **Inventory** directory/directories specified in the **VQT Inventory** sections exist and are empty.

7.4.6.4 Start VQT

1. Click the Start or Start All button to begin the test:



2. If the **Start** button is clicked, only the test highlighted in the list above will be started. If the **Start All** button is clicked, all tests in the list above will be started.

VQT is now in a 'Wait' state, and polling the **Degraded** directory or directories specified in the **VQT Setup**.

When a file is transferred into the **Degraded** directory by VQuad, VQT software will perform comparative testing with the degraded file and the file specified in the **VQT Reference** field of the **VQT Setup**. (Note: This should also be the file transferred by VQuad, as specified in the **Tx Provisioning** of **VQuad Auto-Traffic Configuration**.)

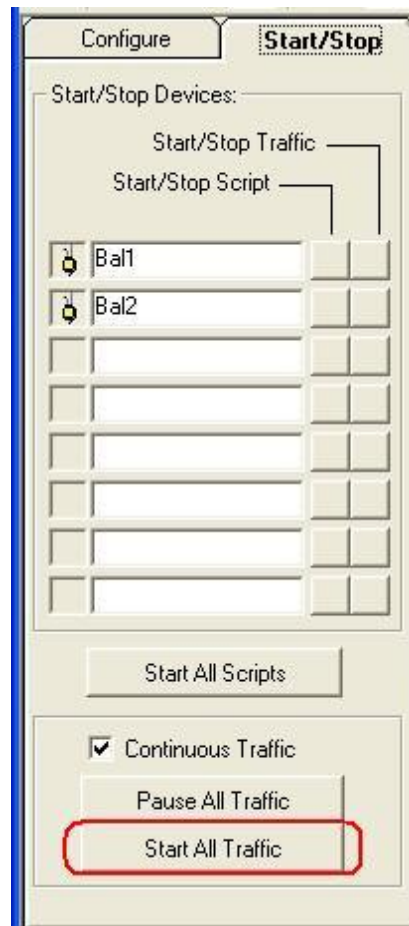
7.4.6.5 Start VQuad:

In the VQuad Main Screen, click the Start All Traffic button:



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3



7.4.7 Transfer/Test Monitoring

7.4.7.1 VQuad Status Monitoring:

The VQuad call and transfer progress can be monitored in two primary locations on the **VQuad Main Screen**:

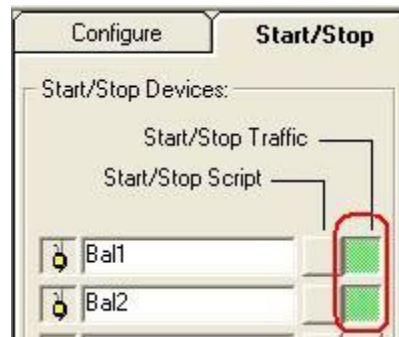
1. Device Status

The upper area of the left panel of the **VQuad Main Screen** will highlight the blocks associated with configured devices when they are active:



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3



2. Call Status

The lower area of the left panel of the **VQuad Main Screen** will indicate the status of calls in progress, and will indicate real-time transmission and receiving activity:

	Call Status	Traffic	Volume	Script
1.	Connected	Running		None
2.	Connected	Running		None

The **Call Status** section will also display activity of the individual devices:

Call Status Section Example 1

	Call Status	Traffic	Volume	Script
1.	Connected	1 Rx File		None
2.	Connected	1 Rx File		None

Call Status Section Example 2

	Call Status	Traffic	Volume	Script
1.	Connected	2 Tx File		None
2.	Connected	Running		None

7.4.7.2 VQT Testing Monitoring

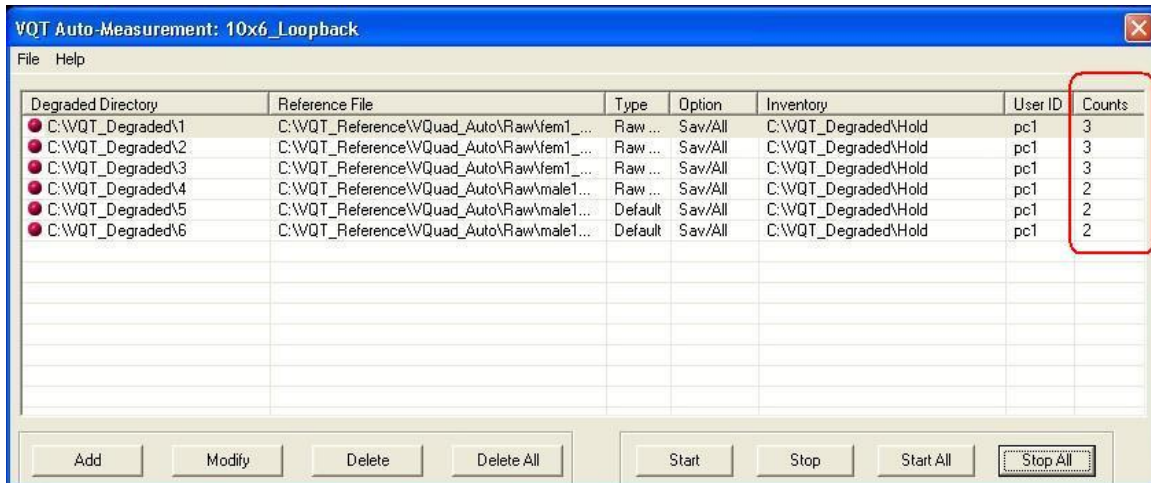
The VQT test and transfer progress can be monitored in two primary locations in VQT:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

1. The Auto Measurement screen will indicate a file count for each running test in the Count column:



The screenshot shows a software window titled "VQT Auto-Measurement: 10x6_Loopback". It contains a table with the following columns: Degraded Directory, Reference File, Type, Option, Inventory, User ID, and Counts. The Counts column is highlighted with a red box. The table lists six entries, each with a file count in the Counts column.

Degraded Directory	Reference File	Type	Option	Inventory	User ID	Counts
C:\VQT_Degraded\1	C:\VQT_Reference\VQuad_Auto\Raw\fm1_...	Raw ...	Sav/All	C:\VQT_Degraded\Hold	pc1	3
C:\VQT_Degraded\2	C:\VQT_Reference\VQuad_Auto\Raw\fm1_...	Raw ...	Sav/All	C:\VQT_Degraded\Hold	pc1	3
C:\VQT_Degraded\3	C:\VQT_Reference\VQuad_Auto\Raw\fm1_...	Raw ...	Sav/All	C:\VQT_Degraded\Hold	pc1	3
C:\VQT_Degraded\4	C:\VQT_Reference\VQuad_Auto\Raw\male1...	Raw ...	Sav/All	C:\VQT_Degraded\Hold	pc1	2
C:\VQT_Degraded\5	C:\VQT_Reference\VQuad_Auto\Raw\male1...	Default	Sav/All	C:\VQT_Degraded\Hold	pc1	2
C:\VQT_Degraded\6	C:\VQT_Reference\VQuad_Auto\Raw\male1...	Default	Sav/All	C:\VQT_Degraded\Hold	pc1	2

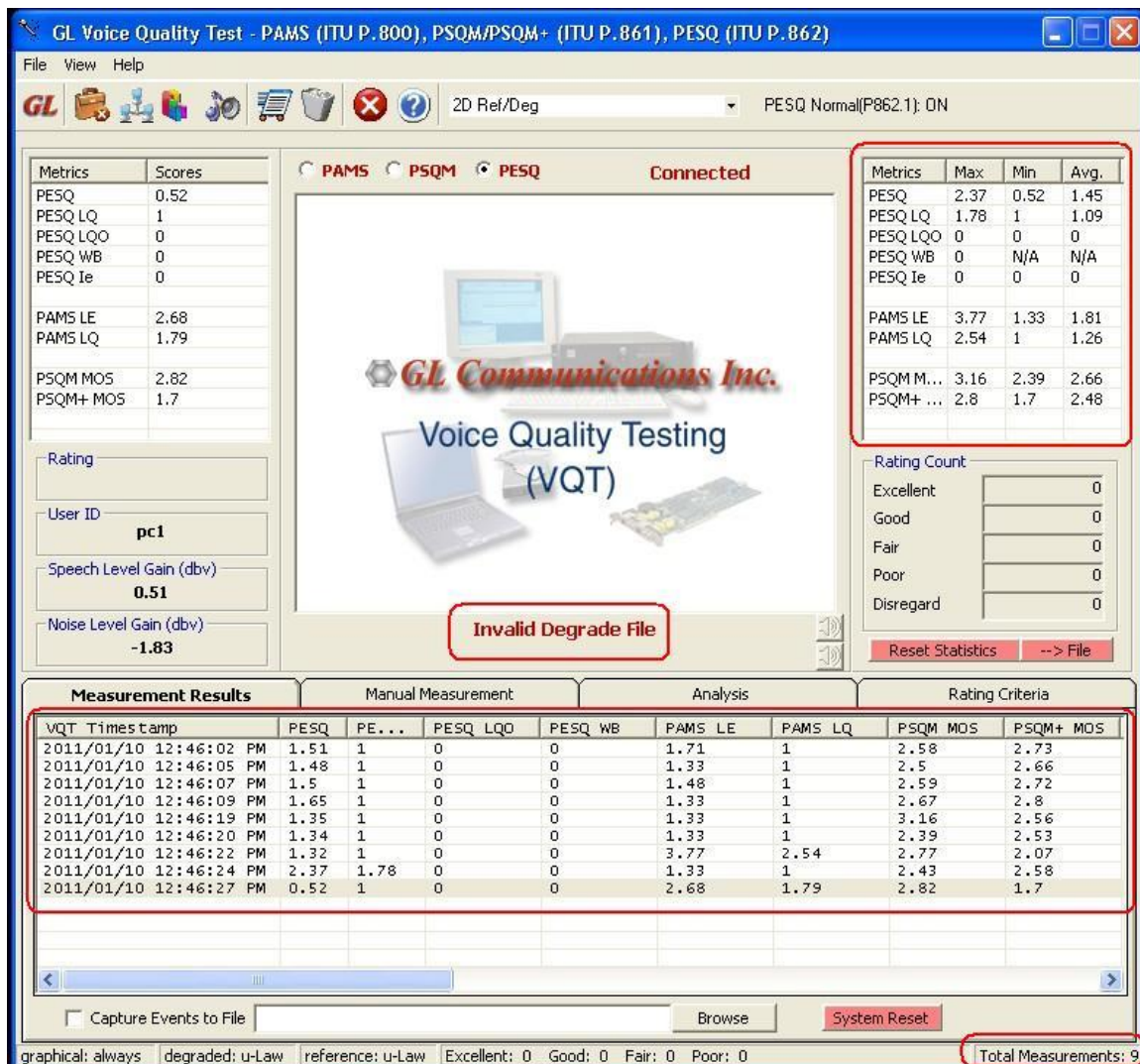
At the bottom of the window, there are buttons for "Add", "Modify", "Delete", "Delete All", "Start", "Stop", "Start All", and "Stop All".

2. The **VQT Main Screen** will also display information about tests in progress, including any alert statuses:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



7.4.7.3 File Movement Monitoring (Windows)

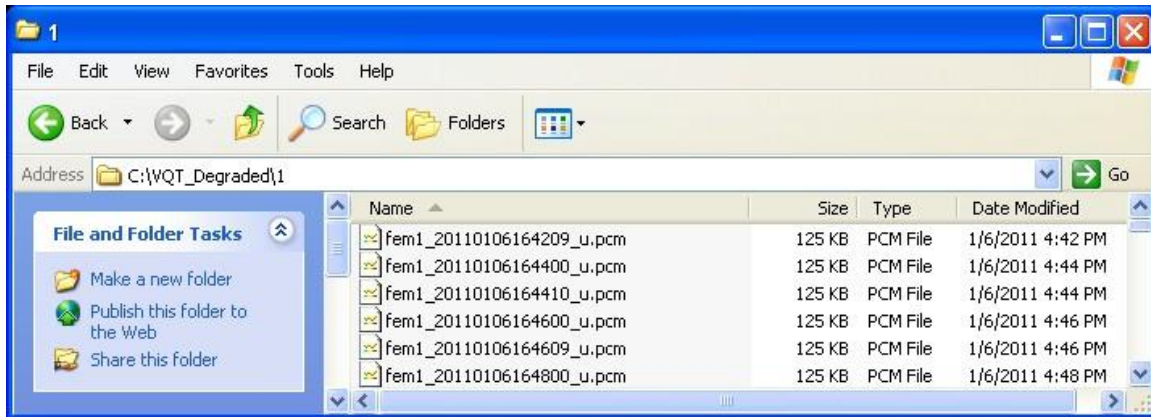
VQuad transfer progress can be monitored externally to VQuad by simply using Windows Explorer (or the Command Window) to view files as they are transferred into the **VQT_Degraded** directory:

- VQT Degraded Directory (Explorer)

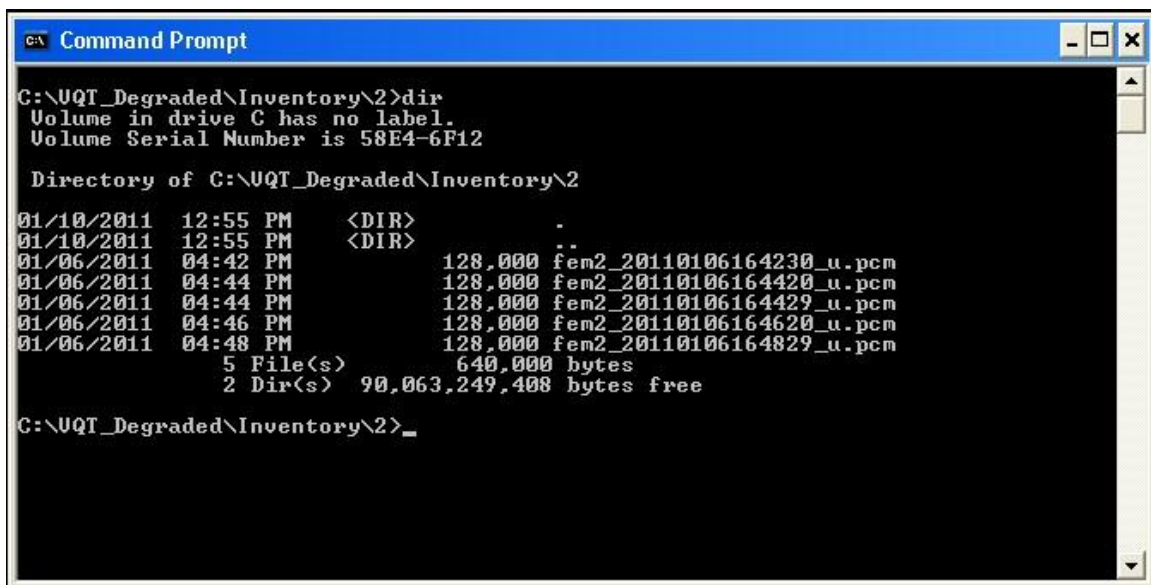


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



b. VQT Degraded Directory (Command Window)

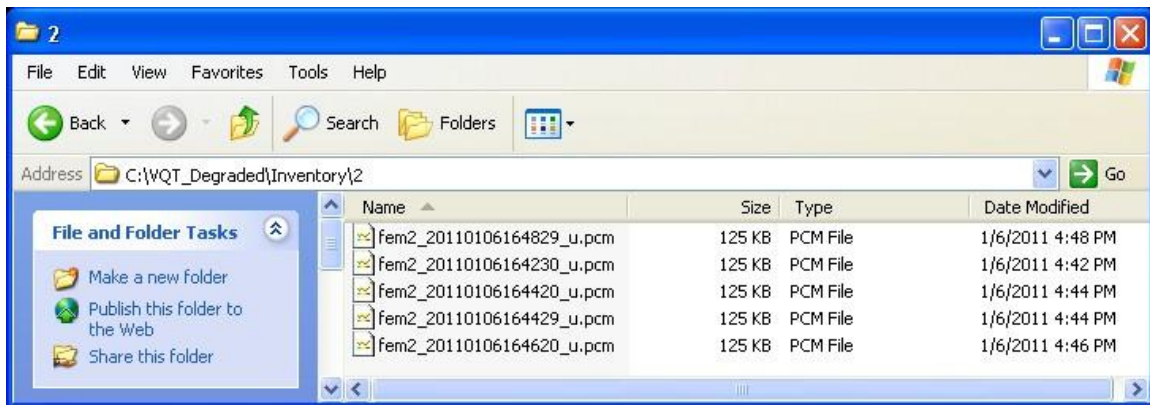


VQT progress can be monitored using Windows Explorer (or the Command Window) to view files as they are transferred from the **VQT_Degraded** directory into the **Inventory** directory:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



Note: If the VQuad transfer and VQT test operations are running simultaneously, the time files remain in the **VQT_Degraded** directory may be very short; depending on circumstances, they may never become visibly listed. (This will occur when VQT testing moves the files from the **VQT_Degraded** before Windows Explorer has updated the listing window.)

Progress can still be monitored, however, by observing the files entering the VQT **Inventory** directory.

7.4.7.4 Stage II: RTP/Transfer Testing

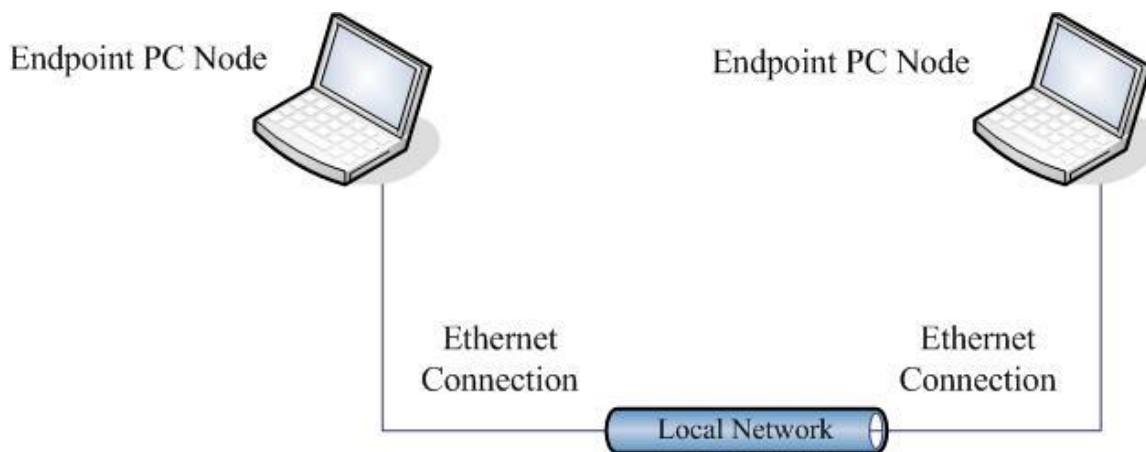


Figure 7-11 PC-to-PC RTP Transfer Test Setup



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

7.4.7.4.1 Detailed Implementation

The baseline for the testing system is defined as the hardware, software, configuration, and files used for testing and verification of the VoIP/SIP solution. Following are the details of the present and planned elements of the baseline.

This implementation tests the Real-Time Protocol (RTP) to be implemented in the integrated solution. It incorporates the files and VQuad functionality of earlier tests, and introduces the element of device-to-device voice file transfer across a network using the developed protocol.

- VoIP/AS-SIP stack (server)
- RTP (endpoint-to-endpoint)
- Endpoint devices

7.4.7.4.2 Hardware Setup

The testing system hardware consists of:

- Two equivalent Dell Latitude D630 laptops as processing nodes
- Connectors and cabling necessary for interconnection

Figure 7-12 illustrates the equipment setup for the above transfers.

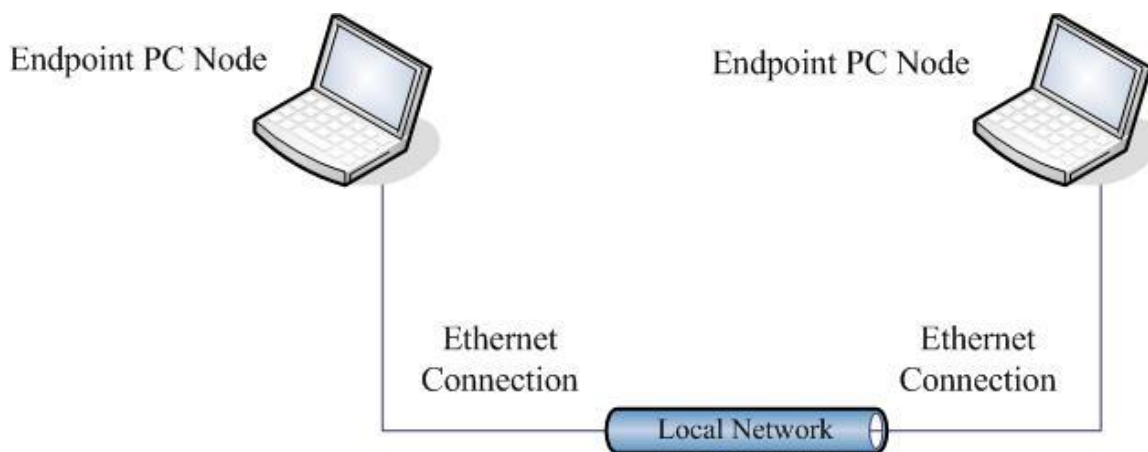


Figure 7-12 PC-to-PC Transfer Test Setup

In addition, each test will be performed with each configured User Agent (UA) in Master and in Slave mode.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

7.4.7.5.3 Two-Endpoint-Node RTP Test Setup:

The configuration used for the RTP tests is a two-PC Endpoint Node system.

To configure hardware for RTP testing:

1. Connect the VoIP Phone Set 1 and VoIP Phone Set 2 network connections to the local Ethernet network via Category 5 cables.

7.4.7.5.4 Data Path

Selected voice files will be transferred from Endpoint PC Node 1 to Endpoint PC Node 2 and stored in the local VQT_Degraded directories. (See Software Setup\VQuad Configuration.)

7.4.7.5.5 Software Setup

The testing and analysis system software consists of:

- Windows XP SP3 (operating system for processing nodes)
- GL Communications VQuad (file transfer and traffic generation software)
- GL Communications VQT (Voice Quality Testing and analysis software)
- Wireshark and other utilities necessary for support and analysis
- Microsoft Excel and Word for data analysis and presentation

7.4.7.5.6 VQuad Setup

To configure GL Communications VQuad for RTP/Transfer Testing:

Use the instructions from **Stage I: Baseline Verification (Loopback Testing) - VQuad Setup**, substituting for Step 6:

From the list of available configurations, select **PCx_Transfer_20x1**.

7.4.7.5.7 Modify or Create a VQuad Device Configuration for RTP Testing

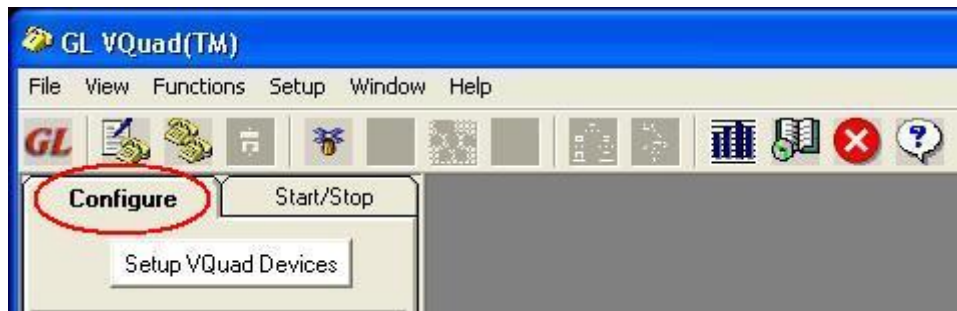
If the predefined configuration is not available, or another configuration is to be created, use the following procedure:

1. Click the **Device Configure** tab:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



2. Click the **Setup VQuad Devices** button:



For RTP testing, two separate devices will be set up as 'VoIP'.

3. In the GL VQuad™ Device Configuration window, click the **Number of Devices** down arrow and select **1**.



4. Click the **Device Type** down arrow and select **VoIP** as the device type:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

GL VQuad(TM) Device Configuration

Number of Devices: Number: 1

Device Configuration: Load Save Delete

	Device Type	Device Name	UTA Type
1.	VoIP	UA1	

Note: Exit VQuad(TM) Software Prior to unPlug Dual UTA

5. Enter a name in the **Device Name** field. For illustration purposes, the **Device Name** entered will be UA1:

GL VQuad(TM) Device Configuration

Number of Devices: Number: 1

Device Configuration: Load Save Delete

	Device Type	Device Name	UTA Type
1.	VoIP	UA1	

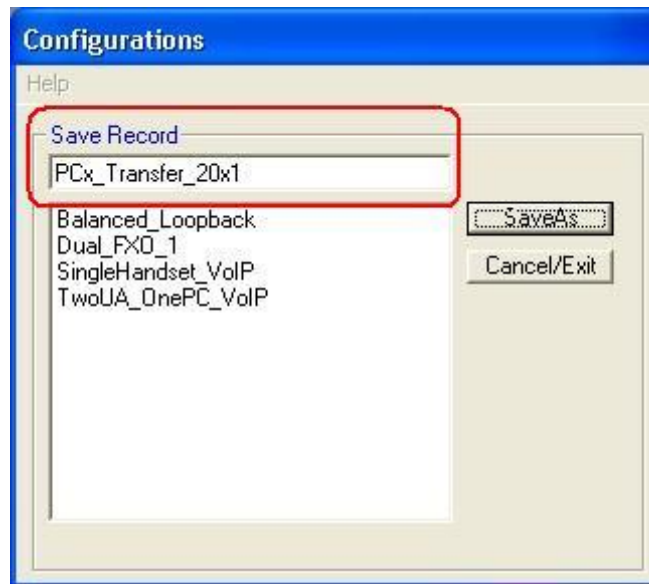
Note: Exit VQuad(TM) Software Prior to unPlug Dual UTA

6. Save the device configuration with a descriptive name using the **Save** button. For illustration purposes, the configuration will be saved as **PCx_Transfer_1**:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

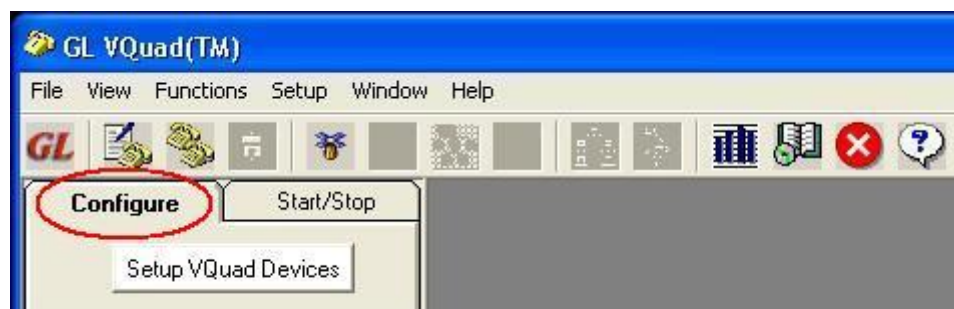


VQuad devices are now configured to transfer files using the configured VoIP end points via Ethernet.

Auto-Traffic Configuration determines the method of transfer (master/slave identification), the type of file transferred, and the locations of files transmitted and received.

Custom Auto-Traffic Configurations have been created for this test implementation. To access and use the configurations, use the following procedure:

In the VQuad Main Window, click the **Configure** tab.





Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

7. To configure **Device 1**, double-click **UA1** (or any other name configured to Device 1 in the previous procedure) in the Devices panel on the left of the VQuad Main Window. Alternately, click the + sign to the left of the named device:



The UA1 detail categories will be listed:



Double-click the **Auto-Traffic** entry:



The **Auto-Traffic Configuration** window will appear:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Auto Traffic Configuration - Auto Created

Device Name: Bal1

☐ Auto Created Traffic Configuration ☒ Use Pre-defined Custom Configuration

Use Pre-defined Custom Configuration

Select Custom Config: 1000x4x2_timed_raw_master1 Show Detail

Configuration: Auto Created Association Device: Bal1

8. Select **UA1** as the **Device Name**:

Auto Traffic Configuration - Auto Created

Device Name: UA1

☐ Auto Created Traffic Configuration ☒ Use Pre-defined Custom Configuration

Use Pre-defined Custom Configuration

Select Custom Config: PCx_Transfer_20x1_Master Show Detail

Configuration: Auto Created Association Device: UA1

9. Click the **Use Pre-defined Custom Configuration** radio button:

Auto Traffic Configuration - Auto Created

Device Name: UA1

☐ Auto Created Traffic Configuration ☒ Use Pre-defined Custom Configuration

Use Pre-defined Custom Configuration

Select Custom Config: PCx_Transfer_20x1_Master Show Detail

Configuration: Auto Created Association Device: UA1

10. Click the down arrow to the right of the **Configuration Name** field to access the list of available configurations.
11. Select the configuration **PCx_Transfer_20x1** for Device 1.
12. Ensure that the following parameters are correct for the relevant device:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'Auto Traffic Configuration - PCx_Transfer_20x1' window with the 'Auto Trigger' tab selected. The 'Master' sub-tab is active. The configuration includes:

- Trigger On Time** (selected):
 - Cycle Time (s): 30
 - Loop Period (s): 120
 - Stop Iterations: 20
 - File Length (s): 8
 - Tx/Rx Offset (s): 6
 - Send Delay (ms): 1500
- Trigger On DTMF digits** (unselected)
- Trigger On MF digits** (unselected)
- Trigger On User Defined Tones** (unselected)

Buttons at the bottom: Load Configuration, Save As Configuration. Status bar: Configuration: PCx_Transfer_20x1, Association Device: UA1.

Figure 7-13 PCx_Transfer_20x1 Auto Trigger Tab

The screenshot shows the 'Auto Traffic Configuration - PCx_Transfer_20x1' window with the 'General' tab selected. The configuration includes:

- File Format:**
 - 8000; 8-bit; A-Law (unselected)
 - 8000; 8-bit; Mu-Law (unselected)
 - 8000; 16-bit; PCM (selected)
- Increment Rx File Name:**
 - Sequential (unselected)
 - Time Stamp (selected)
 - GPS + Time Stamp / ITS + Time Stamp (unselected)
- Digit Parameters:**
 - Enable Digit Tx:**
 - On Time (ms): 200
 - High Power (-dB): 10
 - Enable Multiple Digit Detection:**
 - Off Time (ms): 750
 - Low Power (-dB): 10

Buttons at the bottom: Load Configuration, Save As Configuration. Status bar: Configuration: PCx_Transfer_20x1, Association Device: UA1.

Figure 7-14 PCx_Transfer_20x1 General Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'Tx Provisioning' tab of the 'Auto Traffic Configuration - PCx_Transfer_20x1' window. The window has four tabs: 'Auto Trigger', 'General', 'Tx Provisioning' (selected), and 'Rx Provisioning'. The 'Tx Provisioning' tab contains a table with columns 'Timing' and 'Reference File Path'. The 'Timing' column has values 0, 30, 60, and 90. The 'Reference File Path' column has a single entry 'C:\WQT_Reference\WQuad_Auto\Raw\mem1_1.pcm' for the first row, with empty rows below. To the right of the table are three input fields: 'Cycle Time(ms)' with a value of 2000, 'Identifier Digit' with radio buttons for 'In File' (selected) and 'Generate', 'Power (dB)' with a value of 0, and 'Duration (ms)' with a value of 0. At the bottom, there are buttons for 'Load Configuration' and 'Save As Configuration', and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

Timing	Reference File Path
0	C:\WQT_Reference\WQuad_Auto\Raw\mem1_1.pcm
30	
60	
90	

Figure 7-15 PCx_Transfer_20x1 Tx Provisioning Tab

The screenshot shows the 'Rx Provisioning' tab of the 'Auto Traffic Configuration - PCx_Transfer_20x1' window. The window has four tabs: 'Auto Trigger', 'General', 'Tx Provisioning', and 'Rx Provisioning' (selected). The 'Rx Provisioning' tab contains a table with columns 'Timing' and 'Degraded File Path'. The 'Timing' column has values 8, 38, 68, and 98. The 'Degraded File Path' column has a single entry 'C:\WQT_Degraded\1\mem1' for the first row, with empty rows below. To the right of the table are four input fields, each labeled 'Time (ms)', with values 7000, 7000, 7000, and 7000. At the bottom, there are buttons for 'Load Configuration' and 'Save As Configuration', and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

Timing	Degraded File Path
8	C:\WQT_Degraded\1\mem1
38	
68	
98	

Figure 7-16 PCx_Transfer_20x1 Rx Provisioning Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

7.4.7.5.8 Modify or Create a VQuad Auto Traffic Configuration for RTP Testing

If the existing configuration is not available or is not correct for the required test, use the following procedure to modify it or create a new one:

Configure the **Auto Trigger**, **General**, **Tx Provisioning**, and **Rx Provisioning** tabs in accordance with applicable requirements.

Refer to paragraph 7.4.5.2 Stage I: Baseline Verification (Loopback Testing) - Modify or Create a VQuad Auto Traffic Configuration for guidelines for configuring VQuad fields.

7.4.7.5.9 VQT Setup

Refer to paragraph 7.4.6 Stage I: Baseline Verification (Loopback Testing) - VQT Setup for instructions on configuring VQT to test transferred files.

7.4.7.5.10 Test Execution

With the above configurations in place, the system is prepared to perform automated file transfer and voice file quality testing.

Refer to paragraph 7.4.6 Stage I: Baseline Verification (Loopback Testing) for instructions on starting VQT and VQuad.

7.4.7.5.11 Transfer/Test Monitoring

Refer to paragraph 7.4.7 Stage I: Baseline Verification (Loopback Testing) - Transfer/Test Monitoring for instructions on monitoring system transfers and testing.

7.4.8 Stage III: End-to-End Communication Verification

End-to-End Communication Verification consists of placing a call from an endpoint node (testing the endpoint device and the VoIP/AS-SIP), transferring a voice file (testing the RTP), and ensuring graceful call termination at both endpoints.

This tests and evaluates end-to-end communication of the full integrated solution using selected endpoint nodes.

7.4.9 Detailed Implementation



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The baseline for the testing system is defined as the hardware, software, configuration, and files used for testing and verification of the VoIP/SIP solution. Following are the details of the present and planned elements of the baseline.

7.4.9.1 Hardware Setup

The specific elements tested are:

- Two equivalent Fanstel ST-3116 Phone Sets running L3 software
- RTP transfer between the phone sets

Additional equipment used to perform the tests include:

- One GL Communications Universal Telephony Agent (UTAs) (UTA155155 or UTA155156)
- One Dell Latitude D630 laptop data gathering and processing (PC1 or PC2)
- GL Communications VQuad (running testing PC)
- GL Communications VQT (running testing PC)
- Connectors and cabling necessary for interconnection

The tested operations are:

- Phone Set 1 to Phone Set 2 Voice File Transfer
- Phone Set 2 to Phone Set 1 Voice File Transfer

Figure E1 illustrates the equipment setup for the above transfers.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

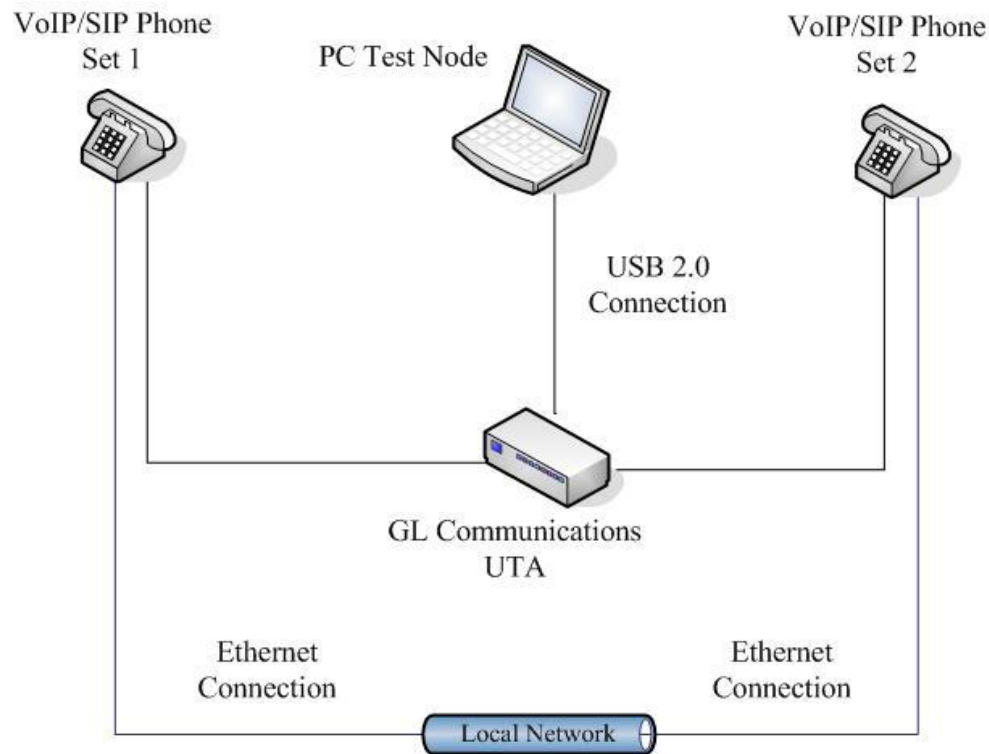


Figure 7-17 SIP Phone – SIP Phone Transfer Test Setup

Note: In the configuration illustrated above, the PC Test Node is not directly involved in call placement or call flow; it is included to record and compile data gathered and produced by the GL Communications UTA.

The testing system hardware consists of:

- Two equivalent Dell Latitude D630 laptops as processing nodes
- One GL Communications Universal Telephony Agent (UTA)
- Two Fanstel ST-3116 phone sets as VoIP/AS-SIP endpoint nodes
- Connectors and cabling necessary for interconnection

To configure hardware for End-to-End testing:

1. Connect the GL Communications UTA to the PC Testing Node via a USB 2.0 connector using the UTA-attached cable.
2. Connect the VoIP Phone Set 1 and VoIP Phone Set 2 network connections to the local Ethernet network via Category 5 cables.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

3. Connect the VoIP Phone Set 1 and VoIP Phone Set 2 handset jacks to the corresponding GL Communications UTA HSet jacks (on the back of the UTA).

7.4.9.2 Data Path

1. Selected voice files will be transferred from the PC Testing Node to Endpoint Phone Set 1 via the UTA.
2. The files will then be transferred via AS-SIP over the Ethernet connections to Endpoint Phone Set 2.
3. The files will then be transferred to the UTA via the headset connection, then to the PC Testing Node for VQT analysis.
4. The transferred files will then be sent to the Endpoint PC Node and stored in the local VQT_Degraded directory. (See Software Setup\VQuad Configuration.)

7.4.9.3 Software Setup

The testing and analysis system software consists of:

- Windows XP SP3 (operating system for processing nodes)
- GL Communications VQuad (file transfer and traffic generation software)
- GL Communications VQT (Voice Quality Testing and analysis software)
- Wireshark and other utilities necessary for support and analysis
- Microsoft Excel and Word for data analysis and presentation

7.4.9.4 VQuad Setup

Use the instructions in paragraph 7.4.6 Stage I: Baseline Verification (Loopback Testing) - VQuad Setup, substituting for Step 6:

From the list of available configurations, select **PCx_EndToEnd_20x1**.

7.4.9.4.1 Modify or Create a VQuad Device Configuration for End-to-End VoIP Testing



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

If the predefined configuration is not available, or another configuration is to be created, use the following procedure:

1. Click the **Device Configure** tab:



2. Click the **Setup VQuad Devices** button:



3. Click the **Number of Devices** down arrow and select **2**.



4. Click the **Device Type** down arrow and select **VoIP** as the device type.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

GL VQuad(TM) Device Configuration

Number of Devices: Number: 1

Device Configuration: Load Save Delete

	Device Type	Device Name	UTA Type
1.	VoIP	UA1	

Note: Exit VQuad(TM) Software Prior to unPlug Dual UTA

5. Enter a name in the **Device Name** field. For illustration purposes, the Device Name entered will be **UA1**:

GL VQuad(TM) Device Configuration

Number of Devices: Number: 1

Device Configuration: Load Save Delete

	Device Type	Device Name	UTA Type
1.	VoIP	UA1	

Note: Exit VQuad(TM) Software Prior to unPlug Dual UTA

6. Save the device configuration with a descriptive name using the **Save Configuration** button. For illustration purposes, the configuration will be saved as **PCx_EndToEnd_1**.

VQuad devices are now configured to transfer files using the configured VoIP end points via the attached UTA.

Auto-Traffic Configuration determines the method of transfer (master/slave identification), the type of file transferred, and the locations of files transmitted and received.

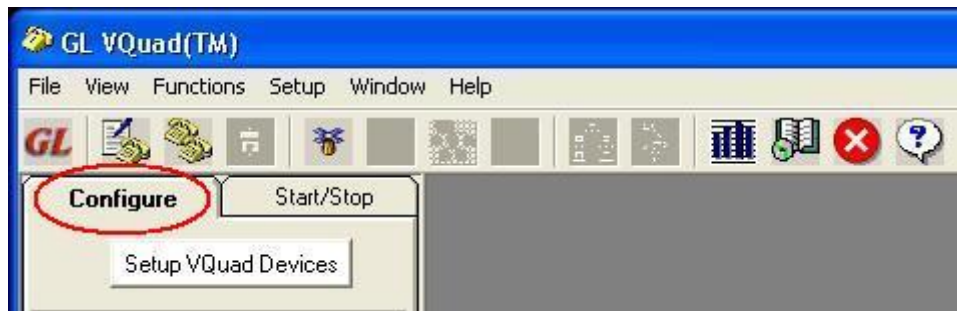
Custom Auto-Traffic Configurations have been created for this test implementation. To access and use the configurations, use the following procedure:

7. In the VQuad Main Window, click the Configure tab.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



8. To configure Device 1, double-click **UA1** (or any other name configured to Device 1 in the previous procedure) in the Devices panel on the left of the VQuad Main Window. Alternately, click the + sign to the left of the named device:



The UA1 detail categories will be listed:



Double-click the **Auto-Traffic** entry:

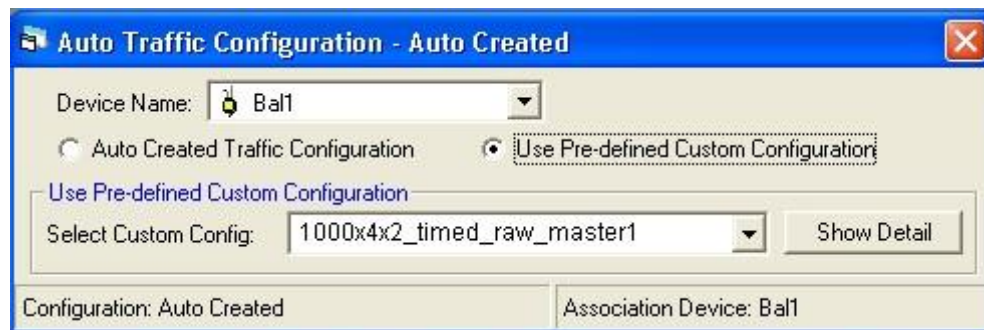


Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3



The **Auto-Traffic Configuration** window will appear:



9. Select **UA1** as the **Device Name**:



10. Click the **Use Pre-defined Custom Configuration** radio button:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



11. Click the down arrow to the right of the **Configuration Name** field to access the list of available configurations.

12. Select the configuration **PCx_EndToEnd_20x1** for Device 1.

Ensure that the following parameters are correct for the relevant device:

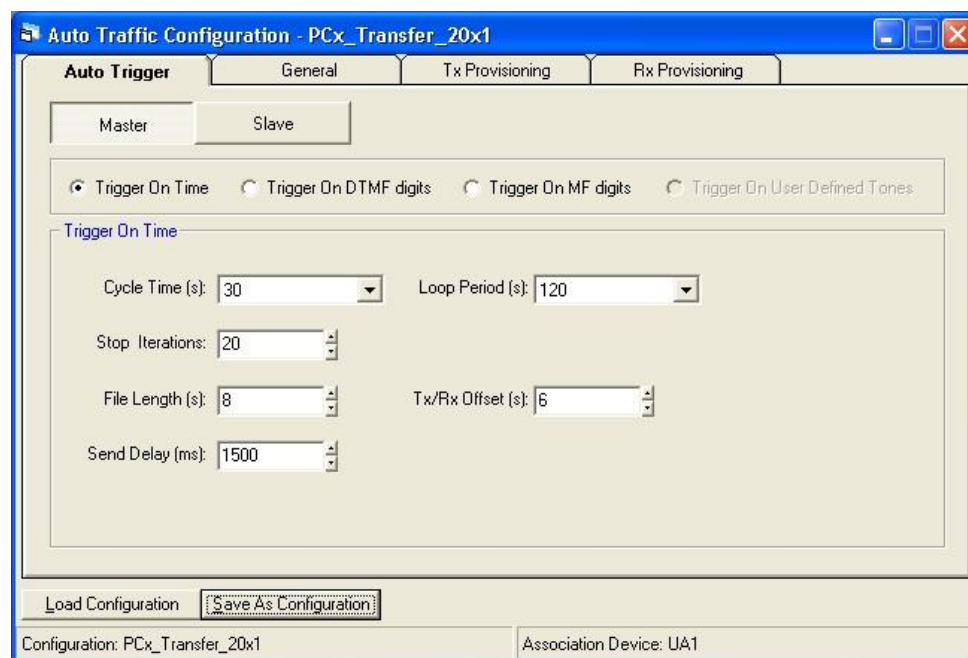


Figure 7-18 PCx_EndToEnd_20x1 Auto-Trigger Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'General' tab of the 'Auto Traffic Configuration - PCx_Transfer_20x1' window. The 'File Format' section has three radio buttons: '8000; 8-bit; A-Law', '8000; 8-bit; Mu-Law', and '8000; 16-bit; PCM', with the last one selected. The 'Increment Rx File Name' section has three radio buttons: 'Sequential', 'Time Stamp', and 'GPS + Time Stamp / ITS + Time Stamp', with 'Time Stamp' selected. The 'Digit Parameters' section has two checkboxes: 'Enable Digit Tx' and 'Enable Multiple Digit Detection', both unchecked. There are four spin boxes: 'On Time (ms)' set to 200, 'Off Time (ms)' set to 750, 'High Power (-dB)' set to 10, and 'Low Power (-dB)' set to 10. At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons, and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

Figure 7-19 PCx_EndToEnd_20x1 General Tab

The screenshot shows the 'Tx Provisioning' tab of the 'Auto Traffic Configuration - PCx_Transfer_20x1' window. The 'Timing' section has a list box with values 0, 30, 60, and 90. The 'Reference File Path' section has a text box containing 'C:\WQT_Reference\WQuad_Auto\Raw\mem1_1.pcm' and a browse button. The 'Cycle Time (ms)' section has a spin box set to 2000. The 'Identifier Digit' section has two radio buttons: 'In File' and 'Generate', with 'In File' selected. The 'Power (dB)' section has a spin box set to 0. The 'Duration (ms)' section has a spin box set to 0. At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons, and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

Figure 7-20 PCx_EndToEnd_20x1 Tx Provisioning Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Timing	Degraded File Path	Time (ms)
8	C:\VQT_Degraded\1\mem1	7000
38		7000
68		7000
98		7000
		7000
		7000
		0
		0
		0
		0

Figure 7-21 PCx_EndToEnd_20x1 Tx Provisioning Tab

7.4.9.4.2 Modify or Create a VQuad Auto Traffic Configuration for End-to-End Testing

If the existing configuration is not available or is not correct for the required test, use the following procedure to modify it or create a new one:

Configure the **Auto Trigger**, **General**, **Tx Provisioning**, and **Rx Provisioning** tabs in accordance with all applicable requirements.

Refer to paragraph 7.4.5.2 Stage I: Baseline Verification (Loopback Testing) - Modify or Create a VQuad Auto Traffic Configuration for guidelines for configuring VQuad fields.

7.4.9.5 VQT Setup

Refer to paragraph 7.4.6 Stage I: Baseline Verification (Loopback Testing) - VQT Setup for instructions on configuring VQT to test transferred files.

7.4.9.5.1 Test Execution

With the above configurations in place, the system is prepared to perform automated file transfer and voice file quality testing.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Refer to paragraph 7.4.5 Stage I: Baseline Verification (Loopback Testing) for instructions on starting VQT and VQuad.

7.4.9.5.2 Transfer/Test Monitoring

Refer to paragraph 7.4.7 Stage I: Baseline Verification (Loopback Testing) - Transfer/Test Monitoring for instructions on monitoring system transfers and testing.

7.5 Summary

The GL Communications products were reviewed in the first Phase of this project. The company is small and working hard to fill several areas and industry needs. However, the last upgrade of their equipment was problem-prone and required large amounts of time to work out issues with the hardware as well as the dongles that are keyed to the individual installation CD for the product. After the issues were resolved the products did perform well and allowed for gathering of the results. There are many other features that we have not explored with the product. It is a combination between software and hardware and requires multiple computers to run efficiently for production readings. It can also be run as a remote system that allows for collection of the data and then post-processing the results afterwards.



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

References

1. *Department of Defense Unified Capabilities Requirements 2008 (UCR 2008), Change 1*
2. *Fusion Voice Engine User's Manual*, Software Version 3.2.X, Part Number/Version: FVE V3.2.X Release Date: November, 2010
3. *GL Communications Voice Quality Testing (VQT) User Guide*, June 2010
4. *GL Communications VQuad™ (VOICE Quad) User Manual*, May 2008
5. *GL Communications GL VQT Reference*, Log Output, September 2004
6. *Packetcable Implementation*, Jeff Riddel, Copyright 2007, Cisco Systems, Inc.



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

CHAPTER 8 Conclusion

8.1 Introduction

The technology is ready for final testing and implementation, but tactical requirements dictate that the implementation is carried out as a structured introduction to the Fleet. The need and direction to have a hybrid system over the last several years has diminished as VoIP in general has become recognized as a stable solution in the industry. However, it must be understood that the tactical readiness of the vessel must be maintained and careful planning and execution of the design of the communications system will be affected by the network that it is placed on. It is our expectation that in the beginning, the network will be a single-use network that will only have communications on it. The network technology for guarding the packets has been designed, and there is agreement on what packets have higher priority (or the bandwidth is so high that it becomes a non - issue) except for denial of service attacks from external or internal sources.

The three Phases of the “Intelligent Advanced Communications IP Telephony Feasibility for the US Navy” for the Office of Naval Research has moved from a paper study that collected white papers from Bell Labs to Avaya (and many others), and created a report that demonstrated how industry was currently using the technology. This technology was not only implemented in small systems but was also being used worldwide, and many of the telephony trunks have already been converted. It allowed for a reduction in cost, weight and size when implemented as a distributed system. The next phase concentrated on the feasibility that Phase I stated, and demonstrated in the lab environment. It also created a Proof-of-Concept of a touch screen tactical communication device that had the beginnings of AS-SIP implementation as we interpreted it from the original UCR before it was released, or had a section for the end device functionality. We contacted US Navy, SPAWAR and NAVSEA individuals and the need for a hybrid changed to an AS-SIP end device. As UCR 2008 Change 1 and the draft of 2 defined it, it could be feasible for a shipboard application and be used for a tactical device. The last phase tackled this question from both an open source and an embedded processor direction. The final result is – yes, it is very feasible for a ship board application, as well as for an administrative and tactical application.

8.2 Conclusion

The test bed has been very dynamic in nature with the addition of a REDCOM SLICE 2100 that was purchased by L-3, in order to foster the Assured Services SIP research. This was done since Avaya did not update the S8300 Switch that we purchased in Phase II for this purpose. It has changed from a network platform testing bed, to a test bed for testing feasibility of AS-SIP end devices with IPv6 and security added since Phase 2 and the removal of the VLANs. It still maintained the purpose of testing feasibility of VoIP on US Navy vessels. The testing performed at L-3 Maritime Systems will allow a design



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

that will compliment the need of the US Navy in forming an IP packet network that will carry voice in the form of an AS-SIP device, as well as data and video into the testing on board a vessel. The project did not design end devices, but instead worked in the feasibility of the end device capability to replace the current telephony that is used on the US Navy vessels.

This project has impacted the future direction of the US Navy in the integration of voice, video and data for vessels. This was shown by the symposiums that its investigators have attended and in its presentations as well as conversations and requests for them to present to individual companies and military organizations. It has shown the feasibility of implementing integration of voice, video and data, in the areas of end device design, AS-SIP variations, IPv6 requirements as well as the need for their integration into a stable network platform.

The merging of the three phases and L-3 Maritime Systems (formerly Henschel) internal research, other US Armed Forces research, doctrines, and rapid advancements in technology, has resulted in a solution that will fulfill the needs of the Fleet. The release of the Unified Capability Requirements 2008 and the follow on Change 2 has brought the direct support for end device Assured-Services SIP, allowing for a complete solution for the US Navy. The newer technologies have been fielded already with the US Army as well as the US Marines. Unified information on the network including communications and critical data will enhance the ability of sailors to meet their responsibilities; and will result as well in the consolidation of devices with which they need to interface. This will also result in reduced training and reduced equipment, thereby reducing space and wiring requirements. While not at their stations, the wireless phones will give the sailor the same comprehensive functionality as a desk phone. The use of open source and open architecture products will keep the US Navy from being tied to a single vendor because of proprietary products. It will allow multiple vendors and their products to be combined into an unified network that will support voice, data and video. This direction is achievable; and it needs to be implemented so that reliability for our sailors is never compromised in any tactical, as well as non-tactical, scenario.



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

APPENDIX A GNU General Public License

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program. To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

* a) The work must carry prominent notices stating that you modified it, and giving a relevant date.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

* b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

* c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

* d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

* a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

* b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

* c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

* d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

* e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

* a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

* b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

* c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

* d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

* e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

* f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License. An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it. A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS (GPL)



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

This page intentionally left blank.



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

APPENDIX B GNU Lesser General Public License

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
 - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
 - 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.
- e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.



APPENDIX C Test Plan

The scope of this test plan includes the interoperability of both AS-SIP and TDM based end instruments (phones) with a REDCOM Slice 2000 SIP switch. Inter-operability testing between multiple switch/carriers falls outside the scope of this test plan. The tests are all centric upon the proper operation of the AS-SIP end instrument only although failures not attributed to the AS-SIP end instrument are also noted. Applicable call flow tests are repeated for both AS-SIP to/from AS-SIP as well as TDM to/from AS-SIP end instruments. User authentication is assumed to be enabled. All tests were repeated for both IPV6 and secure mode. Basic call flow tests were performed with and without MLPP (Multilevel Precedence and Preemption). The nature of SIP required that some tests be verified with a network packet capturing tool. We used both Wireshark and application logging to verify SIP functionality like registration and also to determine the source of any issues encountered.

Test No.	Test Case	Comments
AS001	Register with switch	Verify with Wireshark, also verify appropriate user authentication
AS002	Auto registration	Verify EI is re-registered automatically at least 2 different intervals
AS003	Register with switch, intentionally foul the user authentication password	Verify the EI does not register using Wireshark and or switch admin
AS004	Off hook	Verify audible dial tone, transition to off hook alarm(fast busy) and appropriate display indicator that a line is in use.
AS005	On hook	Verify any line display indicators
AS006	Make call, near end hang up before answer Repeat for both SIP to SIP and SIP to TDM	Verify all audible indicators, dial tone, ringback and call clearing from near on hook. Verify PRACK with Wireshark.
AS007	Make call, near end hang up after answer Repeat for both SIP to SIP and SIP to TDM	Verify all audible indicators, dial tone, ringback and call clearing from near on hook. Verify PRACK with Wireshark.



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Test No.	Test Case	Comments
AS008	Make call, near end holds the call Repeat for both SIP to SIP and SIP to TDM	Verify all audible indicators that the call is on hold
AS009	Make call, near end holds the call and retrieves the call Repeat for both SIP to SIP and SIP to TDM	Verify all audible indicators that the voice path has been re-established
AS010	Make call, far end holds the call Repeat for both SIP to SIP and SIP to TDM	Verify all audible indicators that the call is on hold
AS011	Make call, far end holds the call and retrieves the call Repeat for both SIP to SIP and SIP to TDM	Verify all audible indicators that the voice path has been re-established
AS012	Make call, near end holds the call, near end hangs up Repeat for both SIP to SIP and SIP to TDM	Verify that the held call is unaffected and retrievable
AS013	Make call, near end holds the call, far end hangs up Repeat for both SIP to SIP and SIP to TDM	Verify that the caller phone clears properly, and any visual line indicators are appropriate.
AS014	Make call, far end holds the call, near end hangs up Repeat for both SIP to SIP and SIP to TDM	Verify proper line clearing.
AS015	Make call, far end hangs up after answer Repeat for both SIP to SIP and SIP to TDM	Verify proper line clearing.
AS016	Receive a call no answer Repeat with both SIP and TDM callers	Verify audible ringback and visual line activity indicator
AS017	Receive a call and answer Repeat with both SIP and TDM callers	Verify audio path
AS018	Receive a call and answer, near end on hold Repeat with both SIP and TDM callers	Verify audio path
AS019	Receive a call and answer, near end on hold the retrieve the call Repeat with both SIP and TDM callers	Verify audio path
AS020	Receive a call and answer, far end on hold Repeat with both SIP and TDM callers	Verify audio path



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Test No.	Test Case	Comments
AS021	Receive a call and answer, far end on hold and retrieve the call Repeat with both SIP and TDM callers	Verify audio path
AS022	Make a call and place it on hold/off hold Rapidly at least 6 times finishing in the off hold state	Verify audio path
AS023	Make a call and place it on hold/off hold Rapidly at least 6 times finishing in the on hold state	Verify audio path
AS024	Receive a call and answer, far end on hook Repeat with both SIP and TDM callers	Verify Audio path
AS025	Receive a call and answer, near end on hook Repeat with both SIP and TDM callers	Verify Audio path
AS026	Blind transfer a call from a SIP phone to a SIP phone: SIP station A calls B, B blind transfers A to C. Repeat test hanging up A first, then C first.	Verify phones A&C are connected with a good audio path. Verify station B is no longer connected to either call and now has dial tone if still off hook. Re-verify A&C after B goes on hook.
AS027	Blind transfer a call from a TDM phone to a SIP phone: TDM station A calls SIP station B, B blind transfers A to C. Repeat test hanging up A first, then C first.	Verify phones A&C are connected with a good audio path. Verify station B is no longer connected to either call and now has dial tone if still off hook. Re-verify A&C after B goes on hook.
AS028	Blind transfer a call from a SIP phone to a TDM phone: SIP station A calls TDM station B, B blind transfers A to C. Repeat test hanging up A first, then C first.	Verify phones A&C are connected with a good audio path. Verify station B is no longer connected to either call and now has dial tone if still off hook. Re-verify A&C after B goes on hook.



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Test No.	Test Case	Comments
AS029	Attended transfer a call from a SIP phone to another SIP phone: SIP station A calls SIP station B, B places A on hold and calls C. B takes A off hold.	Verify all 3 parties are now connected with good audio paths.
AS030	Attended transfer a call from a SIP phone to another SIP phone: SIP station A calls SIP station B, B places A on hold and calls C. B takes A off hold. A hangs up first.	Verify audio paths of all 3 parties are correct.
AS031	Attended transfer a call from a SIP phone to another SIP phone: SIP station A calls SIP station B, B places A on hold and calls C. B takes A off hold. B hangs up first.	Verify audio paths of all 3 parties are correct.
AS032	Attended transfer a call from a SIP phone to another SIP phone: SIP station A calls SIP station B, B places A on hold and calls C. B takes A off hold. C hangs up first.	Verify audio paths of all 3 parties are correct.
AS033	Attended transfer a call from a TDM phone to a SIP phone: TDM station A calls SIP station B, B places A on hold and calls C. B takes A off hold.	Verify all 3 parties are now connected with good audio paths.
AS034	Attended transfer a call from a TDM phone to a SIP phone: TDM station A calls SIP station B, B places A on hold and calls C. B takes A off hold. A hangs up first.	Verify audio paths of all 3 parties are correct.
AS035	Attended transfer a call from a TDM phone to a SIP phone: TDM station A calls SIP station B, B places A on hold and calls C. B takes A off hold. B hangs up first.	Verify audio paths of all 3 parties are correct.



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Test No.	Test Case	Comments
AS036	Attended transfer a call from a TDM phone to a SIP phone: TDM station A calls SIP station B, B places A on hold and calls C. B takes A off hold. C hangs up first.	Verify audio paths of all 3 parties are correct.
AS037	Attended transfer a call from a SIP phone to a TDM phone: SIP station A calls TDM station B, B places A on hold and calls C. B takes A off hold.	Verify all 3 parties are now connected with good audio paths.
AS038	Attended transfer a call from a SIP phone to a TDM phone: SIP station A calls TDM station B, B places A on hold and calls C. B takes A off hold. A hangs up first.	Verify audio paths of all 3 parties are correct.
AS039	Attended transfer a call from a SIP phone to a TDM phone: SIP station A calls TDM station B, B places A on hold and calls C. B takes A off hold. B hangs up first.	Verify audio paths of all 3 parties are correct.
AS040	Attended transfer a call from a SIP phone to a TDM phone: SIP station A calls TDM station B, B places A on hold and calls C. B takes A off hold. C hangs up first.	Verify audio paths of all 3 parties are correct.
AS041	Preemption test: set station A's priority to "Routine" via the softkey pad on the phone. Make a single call from station A to station B.	Verify that receiving station(B) is displaying the text version of the incoming priority call: "Routine". No preemption takes place.
AS042	Preemption test: set station A's priority to "Priority" via the softkey pad on the phone. Make a single call from station A to station B.	Verify that receiving station(B) is displaying the text version of the incoming priority call: "Priority". No preemption takes place.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Test No.	Test Case	Comments
AS043	Preemption test: set station A's priority to "Immediate" via the softkey pad on the phone. Make a single call from station A to station B.	Verify that receiving station(B) is displaying the text version of the incoming priority call: "Immediate". No preemption takes place.
AS044	Preemption test: set station A's priority to "FLASH" via the softkey pad on the phone. Make a single call from station A to station B.	Verify that receiving station(B) is displaying the text version of the incoming priority call: "FLASH". No preemption takes place.
AS045	Preemption test: set station A's priority to "FLASH OVERRIDE" via the softkey pad on the phone. Make a single call from station A to station B.	Verify that receiving station(B) is displaying the text version of the incoming priority call: "FLASH OVERRIDE". No preemption takes place.
AS046	Preemption test: Station A makes at "Routine" priority call to station B by prefixing the dialed number with the digits: '94'. Regardless of what the stations priority is set to in the GUI, set the stations priority level to something different than what the dial prefix indicates.	Verify that receiving station(B) is displaying the text version of the incoming priority call: "Routine". No preemption takes place.
AS047	Preemption test: Station A makes at "Priority" priority call to station B by prefixing the dialed number with the digits: '93'. Regardless of what the stations priority is set to in the GUI, set the stations priority level to something different than what the dial prefix indicates.	Verify that receiving station(B) is displaying the text version of the incoming priority call: "Priority". No preemption takes place.
AS048	Preemption test: Station A makes at "Immediate" priority call to station B by prefixing the dialed number with the digits: '92'. Regardless of what the stations priority is set to in the GUI, set the stations priority level to something different than what the dial prefix indicates.	Verify that receiving station(B) is displaying the text version of the incoming priority call: "Immediate". No preemption takes place.
AS049	Preemption test: Station A makes at "FLASH" priority call to station B by prefixing the dialed number with the digits: '91'. Regardless of what the stations priority is set to in the GUI, set the stations priority level to something different than what the dial prefix indicates.	Verify that receiving station(B) is displaying the text version of the incoming priority call: "FLASH". No preemption takes place.



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Test No.	Test Case	Comments
AS050	Preemption test: Station A makes at “FLASH OVERRIDE” priority call to station B by prefixing the dialed number with the digits: ‘90’. Regardless of what the stations priority is set to in the GUI, set the stations priority level to something different than what the dial prefix indicates.	Verify that receiving station(B) is displaying the text version of the incoming priority call: “FLASH OVERRIDE”. No preemption takes place.
AS051	Preemption test: No preemption should occur when the priority domains of calls differ. The resource priority indicator of a call is stored as 3 separate components within the phones software: Example: dsn-000000.6 “DSN” is the network domain. “000000” is the priority domain. “6” is the priority indicator (FLASH) Place 2 Routine priority calls to station A using the default priority domain of “000000”. Place a 3 rd call to station A with a priority domain of “000001” and a the highest priority: FLASH OVERRIDE	Verify the priority domains using Wireshark or any similar network snooping tool. Verify the 3 rd priority call with a priority domain of “000001” and a priority of “FLASH OVERRIDE” was rejected with a busy here and no preemption occurred



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Test No.	Test Case	Comments
AS052	Preemption test: Station A makes a “routine” priority call to station D. Station B makes a “priority” priority call to station D. Station C makes a “FLASH” priority call to station D. All calls have the same priority domain.	<p>Verify that the lowest priority call (station A) receives a continuous preemption tone until the on-hook is received. Station A should also display an indication of the preemption, we display “Preempted”. The called party, station D should also hear the preemption tone for a minimum of 3 seconds and after going on-hook should hear the precedence ring tone and be connected to the preempting caller(station C)</p> <p>Verify, using Wireshark or similar network snooping software, that the preempted caller (station A) also received a “Reason” header on the BYE method set to the following value: Reason: preemption ;cause=1 ;text=“UA Preemption”</p>
AS053	Preemption test: Station B makes a “Routine” priority call to station A and A answers the call. Station A still has one line available. Station C makes a priority call higher than station B to station A.	No preemption should occur here. However, the called party station A should still produce the precedence ring tone. It is entirely up to the called party(A)on how to handle the higher precedence call. As always, the precedence of the incoming call should be displayed on the screen.
AS054	Preemption test: repeat the steps in test AS052 to produce a preemption scenario but make sure that the call to be preempted, the lowest priority call, is currently on hold.	In addition to the checks and verifications of test AS052, verify the held call received the required preemption tone.



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Test No.	Test Case	Comments
AS055	Preemption test: Station B calls station A. Station A answers the call so that it now has one line in use. Station A places station B on hold. Station A now attempts to place an outbound call to station C on the 2 nd line. Station C does not answer the call. Station D makes a priority call (any) to station.	Station A must terminate the incomplete outbound call. Send a BYE or CANCEL with a Reason header of: Reason: preemption ;cause=1 ;text="UA Preemption". There is no reason to play the 3 second preemption tone as the call has not yet been established.
AS056	Call forwarding: station A manually enables call forward "Unconditional" by prefixing the forwarding number with "*72". Example: *725553001. After dialing this forwarding sequence, Use another station to dial station A	Verify that the GUI forwarding status was updated by the manual setting of call forward to unconditional. Verify that the station used to dial station A was forwarded to the desired station, in this example "3001". No preemption should ever take place in unconditional mode.
AS057	Call forwarding: repeat test AS056. Now turn call forward "Unconditional" off by manually dialing "*73". Dial station A from another station.	Verify that the GUI forwarding status was updated by the manual disable of call forward unconditional. GUI screen should now indicate that call forwarding is "OFF". Verify that the station used to dial station A now reaches station A and rings the phone.
AS058	Call forwarding: station A manually enables call forward "No Answer" by prefixing the forwarding number with "*92". Example: *925553001. After dialing this forwarding sequence, Use another station to dial station A	Verify that the GUI forwarding status was updated by the manual setting of call forward to no answer. Verify that the station used to dial station A was forwarded to the desired station, in this example "3001" after a user configurable number of rings(5)



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Test No.	Test Case	Comments
AS059	Call forwarding: repeat test AS058. Now turn call forward “No Answer” off by manually dialing “*93”. Dial station A from another station.	Verify that the GUI forwarding status was updated by the manual disable of call forward no answer. GUI screen should now indicate that call forwarding is “OFF”. Verify that the station used to dial station A now reaches station A and rings the phone.
AS060	Call forwarding: station A manually enables call forward “Busy” by prefixing the forwarding number with “*90”. Example: *905553001. After dialing this forwarding sequence, Use another station to dial station A	Verify that the GUI forwarding status was updated by the manual setting of call forward to busy. Verify that the station used to dial station A was forwarded to the desired station, in this example “3001”
AS061	Call forwarding: repeat test AS060. Now turn call forward “Busy” off by manually dialing “*91”. Dial station A from another station.	Verify that the GUI forwarding status was updated by the manual disable of call forward busy. GUI screen should now indicate that call forwarding is “OFF”. Verify that the station used to dial station A now reaches station A and rings the phone.
	Call forwarding: repeat tests AS060, AS058 and AS056 except use the GUI interface softkey to cycle through all 4 call forwarding states: OFF, BUSY, NO ANSWER, UNCONDITIONAL.	



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

APPENDIX D End-to-End Test Communication Verification



Table of Contents

APPENDIX D	END-TO-END TEST COMMUNICATION VERIFICATION	233
D-1	END-TO-END COMMUNICATION VERIFICATION	235
D-2	DETAILED IMPLEMENTATION	235
D-3	HARDWARE SETUP	235
D-3.1	<i>Data Path</i>	237
D-4	SOFTWARE SETUP	237
D-4.1	<i>VQuad Setup</i>	237
D-4.2	<i>Device Configuration</i>	238
D-4.3	<i>Modify or Create a VQuad Device Configuration for End-to-End VoIP Testing</i>	239
D-4.4	<i>Auto-Traffic Configuration</i>	241
D-4.6	<i>Modify or Create a VQuad Auto Traffic Configuration for End-to-End Testing</i>	246
D-5	VQT SETUP	248
D-6	CHECKING VQT AUTO-MEASUREMENT SETUP	254
D-7	TEST EXECUTION	256
D-7.1	<i>Preparation</i>	256
D-7.2	<i>Start VQT</i>	257
D-7.3	<i>Start VQuad</i>	257
D-7.4	<i>Transfer/Test Monitoring</i>	258
D-7.5	<i>VQT Testing Status</i>	260
D-7.6	<i>File Movement Monitoring (Windows)</i>	261
D-8	TEST RESULTS	263
D-8.1	<i>PAMS</i>	264
D-8.2	<i>PSQM and PSQM+</i>	264
D-8.3	<i>PESQ</i>	265
D-8.4	TEST FILE VERIFICATION/AUDITING	265



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

D-1 End-to-End Communication Verification

End-to-End Communication Verification consists of placing a call from an endpoint node (testing the endpoint device and the VoIP/AS-SIP), transferring a voice file (testing the RTP), and ensuring graceful call termination at both endpoints.

This tests and evaluates end-to-end communication of the full integrated solution using selected endpoint nodes. The elements under test are:

D-2 Detailed Implementation

The baseline for the testing system is defined as the hardware, software, configuration, and files used for testing and verification of the VoIP/SIP solution. Following are the details of the present and planned elements of the baseline.

D-3 Hardware Setup

The specific elements tested are:

- Two equivalent Fanstel ST-3116 Phone Sets running L3 software
- RTP transfer between the phone sets

Additional equipment used to perform the tests include:

- One GL Communications Universal Telephony Agent (UTAs) (UTA155155 or UTA155156)
- One Dell Latitude D630 laptop data gathering and processing (PC1 or PC2)
- GL Communications VQuad (running testing PC)
- GL Communications VQT (running testing PC)
- Connectors and cabling necessary for interconnection

The tested operations are:

- Phone Set 1 to Phone Set 2 Voice File Transfer
- Phone Set 2 to Phone Set 1 Voice File Transfer

Figure D-1 illustrates the equipment setup for the above transfers.



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

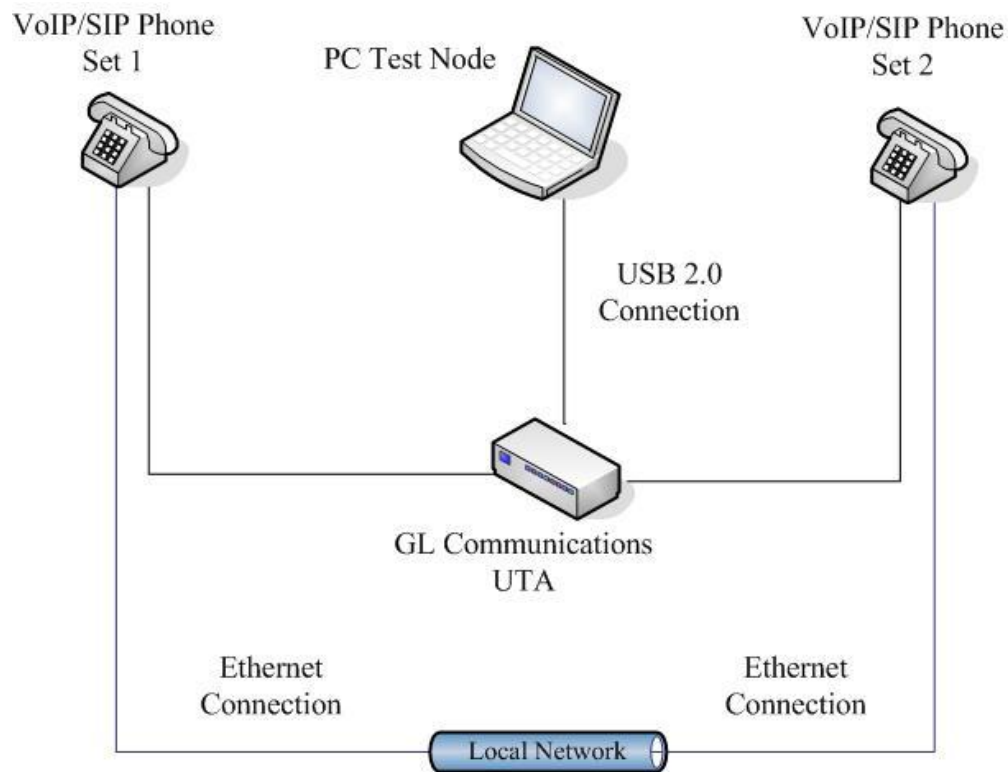


Figure D-1 SIP Phone – SIP Phone Transfer Test Setup

The testing system hardware consists of:

- Two equivalent Dell Latitude D630 laptops as processing nodes
- One GL Communications Universal Telephony Agent (UTA)
- Two Fanstel ST-3116 phone sets as VoIP/AS-SIP endpoint nodes
- Connectors and cabling necessary for interconnection

To configure hardware for End-to-End testing:

2. Connect the GL Communications UTA to the PC Testing Node via a USB 2.0 connector using the UTA-attached cable.
3. Connect the VoIP Phone Set 1 and VoIP Phone Set 2 network connections to the local Ethernet network via Category 5 cables.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

3. Connect the VoIP Phone Set 1 and VoIP Phone Set 2 handset jacks to the corresponding GL Communications UTA HSet jacks (on the back of the UTA).

D-3.1 Data Path

5. Selected voice files will be transferred from the PC Testing Node to Endpoint Phone Set 1 via the UTA.
6. The files will then be transferred via AS-SIP over the Ethernet connections to Endpoint Phone Set 2.
7. The files will then be transferred to the UTA via the headset connection, then to the PC Testing Node for VQT analysis.
8. The transferred files will then be sent to the Endpoint PC Node and stored in the local VQT_Degraded directory. (See Software Setup\VQuad Configuration.)

D-4 Software Setup

The testing and analysis system software consists of:

- Windows XP SP3 (operating system for processing nodes)
- GL Communications VQuad (file transfer and traffic generation software)
- GL Communications VQT (Voice Quality Testing and analysis software)
- Wireshark and other utilities necessary for support and analysis
- Microsoft Excel and Word for data analysis and presentation

D-4.1 VQuad Setup

To configure GL Communications VQuad for End-to-End Communication Verification:

7. Start VQuad by double-clicking the GL VQuad icon:

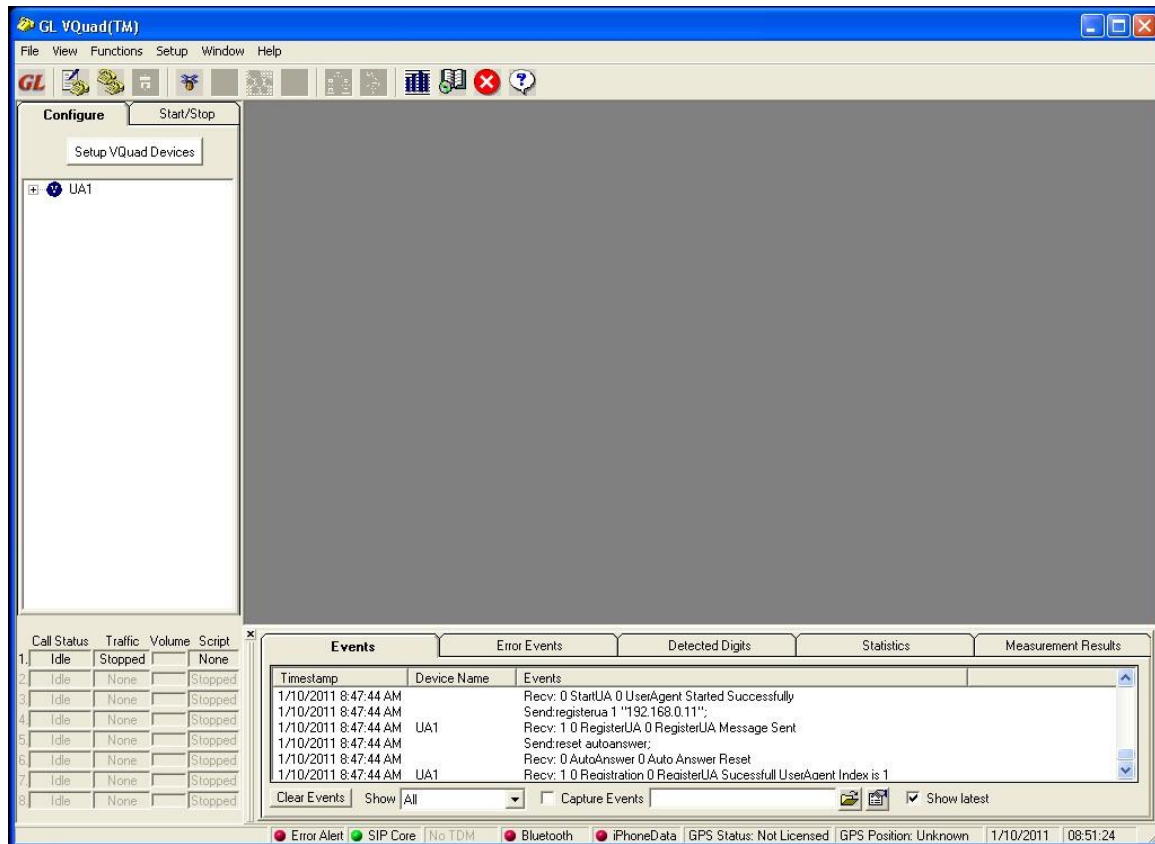


8. The VQuad Main Screen appears.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



D-4.2 Device Configuration

A custom configuration has been created for this test implementation. To access and use the configuration, click the **Configuration** button and select **PCx_EndToEnd_20x1**.

The naming scheme for device configurations is as follows:

For the configuration **PCx_EndToEnd_20x1**,

PCx_ indicates the PC testing node for which the configuration is designed:

- **PC1** indicates Endpoint PC Node 1
- **PC2** indicates Endpoint PC Node 2
- **PCx** indicates either PC Node can be used for the configuration

EndToEnd_ indicates the type of test:

- **Loopback** indicates the test is single-node, endpoint-to-same-endpoint



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- **Transfer** indicates the test is multi-node (two or more nodes), transfer endpoints different
- **EndToEnd** indicates that all devices in a VoIP/AS-SIP communication path will be tested

20x1 indicates the number of transfers and the number of files:

- The first number (in this case, **20**) is the total number of transfers to be performed
- The second number (in this case, **1**) is the number of files which will be transferred

Thus, the configuration name **PC2_Transfer_100x4** would indicate:

- The configuration is designed for PC2
- It is a multi-node transfer
- 100 transfers of four files will be performed

D-4.3 Modify or Create a VQuad Device Configuration for End-to-End VoIP Testing

If the predefined configuration is not available, or another is required to be created, use the following procedure:

If the predefined configuration is not available, or another configuration is to be created, use the following procedure:

1. Click the **Device Configure** tab:



2. Click the **Setup VQuad Devices** button:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

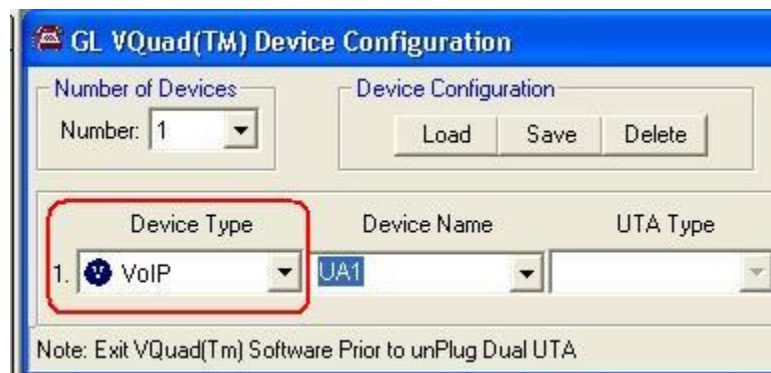
Maritime Systems



- Click the **Number of Devices** down arrow and select **2**.



- Click the **Device Type** down arrow and select **VoIP** as the device type.

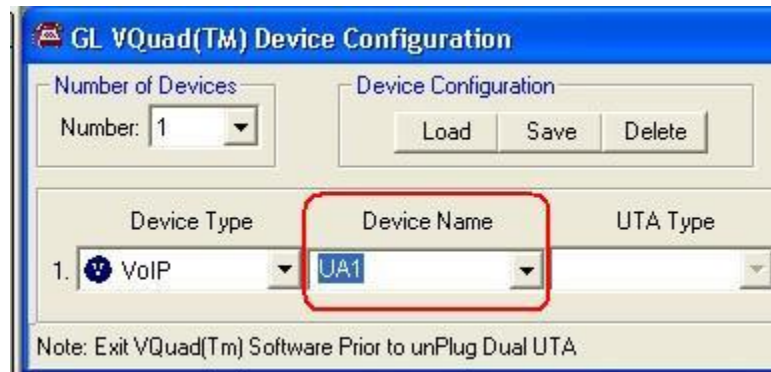


- Enter a name in the **Device Name** field. For illustration purposes, the Device Name entered will be **UA1**:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



6. Save the device configuration with a descriptive name using the **Save Configuration** button. For illustration purposes, the configuration will be saved as **PCx_EndToEnd_1**.

VQuad devices are now configured to transfer files using the configured VoIP end points via the attached UTA.

D-4.4 Auto-Traffic Configuration

Auto-Traffic Configuration determines the method of transfer (master/slave identification), the type of file transferred, and the locations of files transmitted and received.

Custom Auto-Traffic Configurations have been created for this test implementation. To access and use the configurations, use the following procedure:

1. In the VQuad Main Window, click the Configure tab.





Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

D-4.5 Configure Device 1

1. Double-click **UA1** (or any other name configured to Device 1 in the previous procedure) in the Devices panel on the left of the VQuad Main Window. Alternately, click the + sign to the left of the named device:



The UA1 detail categories will be listed:



Double-click the **Auto-Traffic** entry:



The **Auto-Traffic Configuration** window will appear:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Auto Traffic Configuration - Auto Created

Device Name: Bal1

☐ Auto Created Traffic Configuration ☒ Use Pre-defined Custom Configuration

Use Pre-defined Custom Configuration

Select Custom Config: 1000x4x2_timed_raw_master1 Show Detail

Configuration: Auto Created Association Device: Bal1

2. Select **UA1** as the **Device Name**:

Auto Traffic Configuration - Auto Created

Device Name: UA1

☐ Auto Created Traffic Configuration ☒ Use Pre-defined Custom Configuration

Use Pre-defined Custom Configuration

Select Custom Config: PCx_Transfer_20x1_Master Show Detail

Configuration: Auto Created Association Device: UA1

3. Click the **Use Pre-defined Custom Configuration** radio button:

Auto Traffic Configuration - Auto Created

Device Name: UA1

☐ Auto Created Traffic Configuration ☒ Use Pre-defined Custom Configuration

Use Pre-defined Custom Configuration

Select Custom Config: PCx_Transfer_20x1_Master Show Detail

Configuration: Auto Created Association Device: UA1

4. Click the down arrow to the right of the **Configuration Name** field to access the list of available configurations.
5. Select the configuration **PCx_EndToEnd_20x1** for Device 1.

Ensure that the following parameters are correct for the relevant device:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'Auto Traffic Configuration - PCx_Transfer_20x1' window with the 'Auto Trigger' tab selected. The 'Master' sub-tab is active. Under 'Trigger On Time', the 'Cycle Time (s)' is set to 30, 'Loop Period (s)' to 120, 'Stop Iterations' to 20, 'File Length (s)' to 8, 'Tx/Rx Offset (s)' to 6, and 'Send Delay (ms)' to 1500. The 'Trigger On DTMF digits', 'Trigger On MF digits', and 'Trigger On User Defined Tones' options are disabled. At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons, and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

PCx_EndToEnd_20x1 Auto-Trigger Tab

The screenshot shows the 'Auto Traffic Configuration - PCx_Transfer_20x1' window with the 'General' tab selected. Under 'File Format', the '8000; 16-bit; PCM' option is selected. Under 'Increment Rx File Name', the 'Time Stamp' option is selected. In the 'Digit Parameters' section, 'Enable Digit Tx' is checked with 'On Time (ms)' set to 200 and 'High Power (-dB)' set to 10. 'Enable Multiple Digit Detection' is checked with 'Off Time (ms)' set to 750 and 'Low Power (-dB)' set to 10. At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons, and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

PCx_EndToEnd_20x1 General Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'Tx Provisioning' tab of the 'Auto Traffic Configuration - PCx_Transfer_20x1' window. The window has four tabs: 'Auto Trigger', 'General', 'Tx Provisioning' (selected), and 'Rx Provisioning'. The 'Tx Provisioning' tab contains a 'Timing' table with values 0, 30, 60, and 90. To the right is a 'Reference File Path' table with one row containing 'C:\WQT_Reference\WQuad_Auto\Raw\mem1_1.pcm'. Further right are input fields for 'Cycle Time(ms)' (2000), 'Identifier Digit' (radio buttons for 'In File' and 'Generate'), 'Power (dB)' (0), and 'Duration (ms)' (0). At the bottom are 'Load Configuration' and 'Save As Configuration' buttons, and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

Timing	Reference File Path	Cycle Time(ms)	Identifier Digit	Power (dB)	Duration (ms)
0	C:\WQT_Reference\WQuad_Auto\Raw\mem1_1.pcm	2000	<input checked="" type="radio"/> In File <input type="radio"/> Generate	0	0
30					
60					
90					

PCx_EndToEnd_20x1 Tx Provisioning Tab

The screenshot shows the 'Rx Provisioning' tab of the 'Auto Traffic Configuration - PCx_Transfer_20x1' window. The window has four tabs: 'Auto Trigger', 'General', 'Tx Provisioning', and 'Rx Provisioning' (selected). The 'Rx Provisioning' tab contains a 'Timing' table with values 8, 38, 68, and 98. To the right is a 'Degraded File Path' table with one row containing 'C:\WQT_Degraded\1\mem1'. Further right is a 'Time (ms)' table with eight rows, the first four containing 7000 and the last four containing 0. At the bottom are 'Load Configuration' and 'Save As Configuration' buttons, and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

Timing	Degraded File Path	Time (ms)
8	C:\WQT_Degraded\1\mem1	7000
38		7000
68		7000
98		7000
		7000
		7000
		0
		0
		0
		0

PCx_EndToEnd_20x1 Tx Provisioning Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

D-4.6 Modify or Create a VQuad Auto Traffic Configuration for End-to-End Testing

If the existing configuration is not available or is not correct for the required test, use the following procedure to modify it or create a new one:

Configure the **Auto Trigger**, **General**, **Tx Provisioning**, and **Rx Provisioning** tabs in accordance with all applicable requirements.

Guidelines for configuring the fields of the tabs follow:

D-4.6.1 Auto Trigger Tab Fields

- **Cycle Time:** This is the time allotted for the VQuad™ master and slave operations to take place. This time should be adequately long when bidirectional operations are performed.
- **Loop Period:** This is the total length of a timed cycle. (The default profile uses six files in a cycle.)
- **Stop Iteration:** Specifies the number of iterations (cycles multiplied by number of lines configured) that the VQuad™ will execute before stopping the test.
- **File Length:** This indicates the recording duration and is required to set up the VQuad™ time triggering flow. This includes the time to receive an entire file length plus the send delay time.
- **Tx/Rx Offset:** This is the ‘wait period’ after completion of a Tx action, but before the return of Tx action for bi-directional file transfer.
- **Send Delay:** This is the time after which the VQuad™ receiving end starts recording. This is a buffer time which helps counter the clock variations of two independent PCs.

D-4.6.2 General Tab Fields

- **File Format:** This setting selects the format of the file that the VQuad™ receiver will be saved to after the receiver completes recording of an incoming degraded file. The recorded degraded file type must match the type of VQuad™ reference file being sent, or the VQT Analysis program will yield very bad VQT scores due to mismatched file types. VQuad™ reference files are provided in linear PCM, A-



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

law, and Mu-law encoding formats. There are usually small differences in VQT scores when using one file type as opposed to another. VQuad™ normally executes using linear PCM Reference files.

- Note: When performing transfers using Dual UTAs, the file type must be 16-bit 8000 Hz sampled, Little Endian (Intel) PCM.
- Increment Rx File Name : When the VQuad™ completes recording of an incoming degraded voice file, the file several options can be used to suffix the received file name to simplify identifying the degraded files.

The filename prefix for each recorded sample is defined by the filename rules setup in the Rx Provisioning tab.

- Sequential: The sequential option simply appends a sequential number to each sample filename in the order as they are received. By default, the numbering starts from zero, but the user can select Sequential File Numbering to start the numbering sequence at any number, or to reset a sequence that was previously started.
- Time Stamp: Each incoming/recorded-degraded file has the time stamp (from the PC clock) that the sample was received as the filename suffix. (Note: Ensure that the PC clock is set to the correct time.)
- Digit Parameters
 - The Digit/Tone Parameters only work in conjunction with the User-defined Tone Triggering method when trigger tones external to the sample voice file are required.
 - If Enable Digit TX is not checked, the VQuad™ program uses MF, DTMF or User-defined tones, attached (prefixed) to the VQT Reference File, to trigger the recording and to identify the VQT Reference File that follows. If Enable Digit TX is checked, the VQuad™ program sends tones according to the table of user-defined tones before sending voice file. In this case, the reference voice files without prefixed tones must be used. (Refer to section Tone/Digit Method for Sending/Recording for details)
 - Enable Multiple Digit Detection is used only in 'master mode' with DTMF/MF trigger. It allows multiple digits detection during one session.
 - For example, in normal case (Enable Multiple Digit Detection is not checked), one session consists of a 'TX file' and a 'RX file'. Even if it detects another digit, it will wait till the end of the cycle duration and then proceeds to the next session. However, the Enable Multiple Digit Detection option when checked, will again allow "RX" file event to occur after another digit is detected.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- Enter On Time (200ms or longer), Off Time (750ms or longer), High Power (-dB) and Low Power (-dB) values.

D-4.6.3 Tx Provisioning Tab Fields

- Reference File Path: This field determines the location of the file(s) to be transferred.

Sample files are provided by GL Communications, and are located in local subdirectories of C:\VQT_Reference\.

D-4.6.4 Rx Provisioning Tab Fields

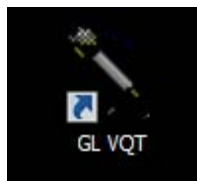
- Degraded File Path: The path on the local PC which is the destination for files transferred in the assigned time slot.

The final characters of the **Degraded File Path** entry will become the initial characters of the files placed in the Degraded File Path directory. These characters will be suffixed with the **Sequential** or **Time Stamp** data (configured in the **General** tab).

D-5 VQT Setup

To configure GL Communications VQT for Auto Measurement Setup and Execution:

1. Start VQT by double-clicking the GL Voice Quality Testing icon:

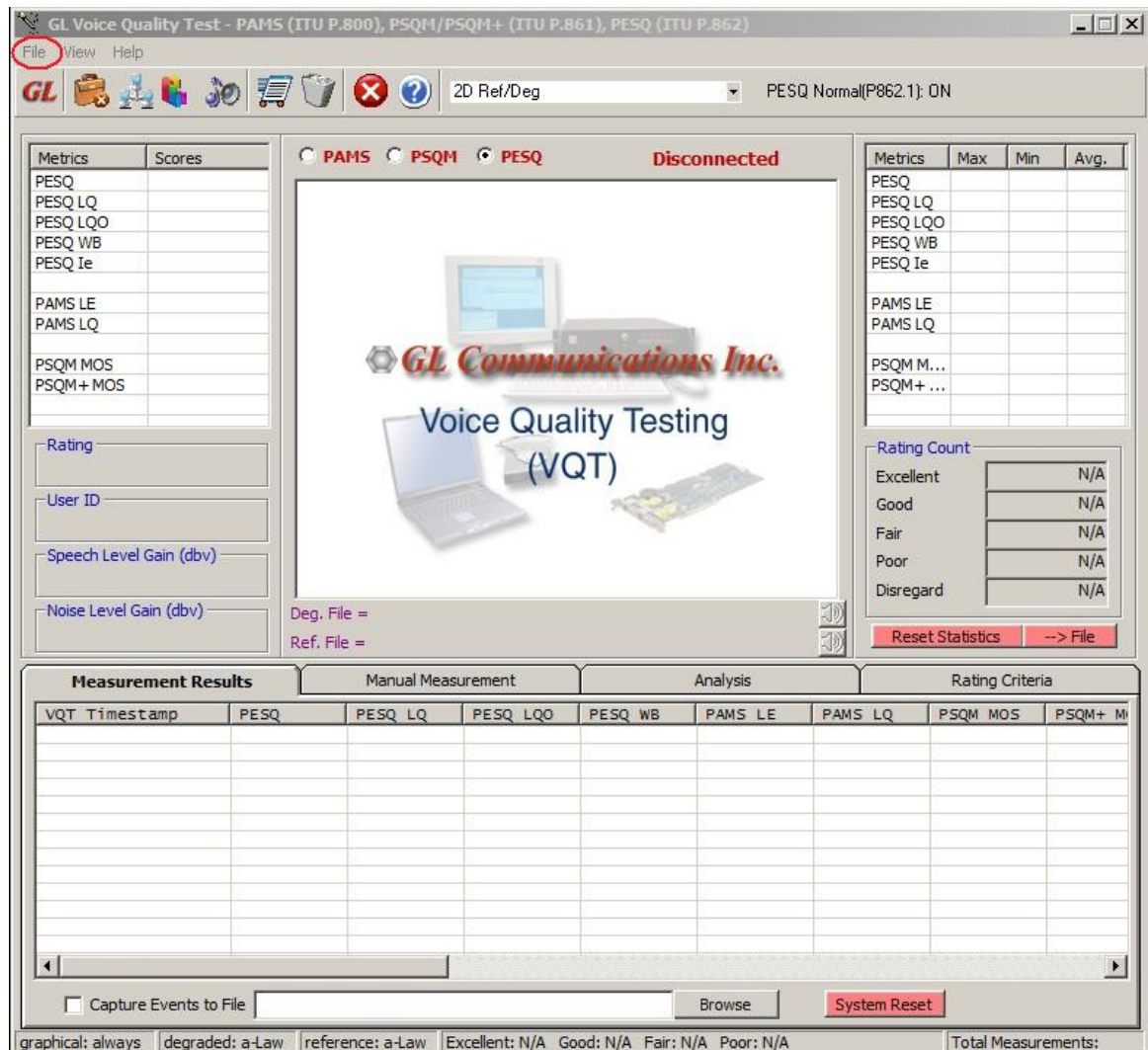


2. The VQT Main Screen appears.

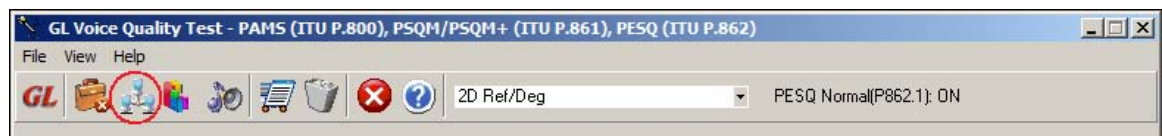


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



4. Alternatively, the **Auto-Measurement** icon can be clicked:



5. The **VQT Auto-Measurement** screen appears:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

8. Enter the **Degraded Directory** in the space provided. This is the directory containing the files transferred in the VQuad RTP Test, and is configured in VQuad Auto Traffic Configuration. (See **VQuad Auto Traffic Configuration – Rx Provisioning Tab.**)

9. For illustration purposes, **C:\VQT_Degraded\1** will be entered.

Note: Ensure that the directory **C:\VQT_Degraded\1** (or whatever directory is specified as the **Degraded Directory**) exists on the hard drive. Clicking the **Open Folder** icon will open a Windows selection dialogue which will allow selection from existing directories:

10. Enter the **Reference File** in the space provided. This is the original file which was transferred according the VQuad RTP Test, and is configured in VQuad Auto



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Traffic Configuration. (See **VQuad Auto Traffic Configuration – Tx Provisioning Tab.**)

11. For illustration purposes, **C:\VQT_Reference\VQuad_Auto\fem1.pcm** will be selected.

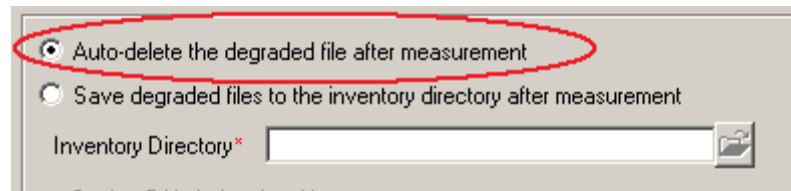
12. Select the **Save degraded files to an inventory directory after measurement** radio button.

13. Enter the **Inventory Directory** in the space provided. This is the directory to which files will be transferred after testing. Thus, the file path is:

14. For illustration purposes **C:\VQT_Degraded\Inventory\1** will be selected.

Note: Ensure that **C:\VQT_Degraded\1** exists on the hard drive. Clicking the Open Folder icon will open a Windows selection dialogue which will allow selection from existing directories.

Alternately, the **Auto-delete the degraded file after measurement** radio button can be selected:



If this button is selected, the transferred files will be deleted from the Degraded directory after testing. This option may be preferred if the test parameters are known correct, and the test measurements will not be repeated.

However, for purposes of data retention, or if running any given tests again is necessary or possible, the use of the **Save degraded files to an inventory directory after measurement** is preferable.

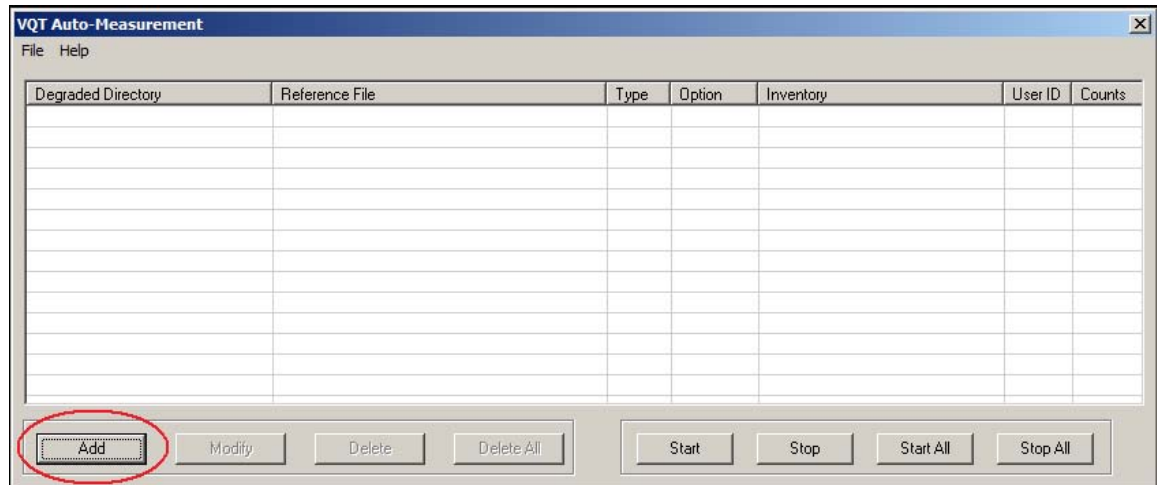
15. Within the **User ID** field, enter a User ID for tracking purposes. This field is not verified or cross-checked; it is stored as part of the data produced by the test, and can be useful for granular data analysis in a multi-testing environment.

16. Select the **Add** button to add the session

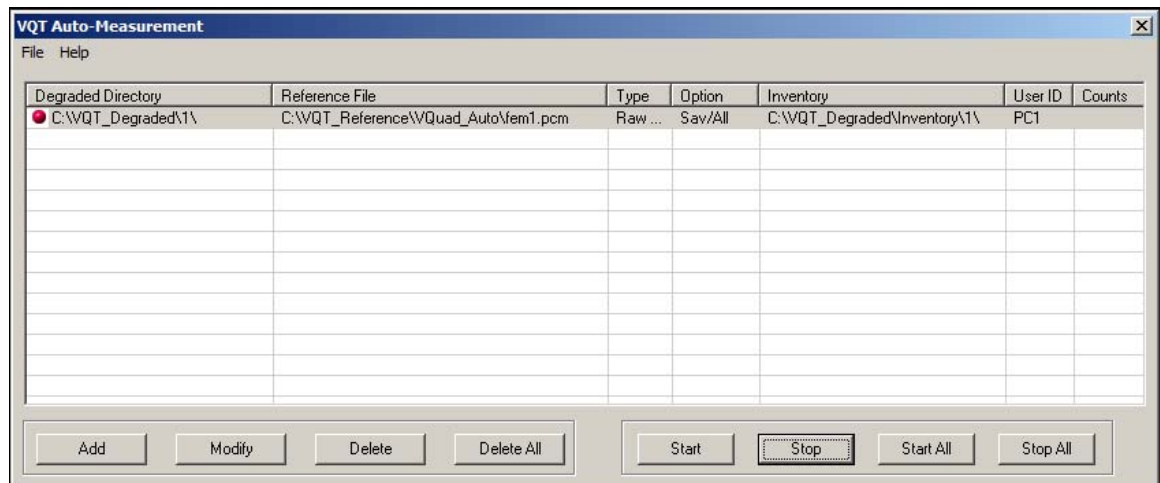


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



17. The **VQT Auto-Measurement** window will now indicate the data associated with the test entered:



D-6 Checking VQT Auto-Measurement Setup

To verify that the VQT setup is functional, manually copy a file of the correct format into the specified Degraded directory. If setup is correct, VQT will automatically run the measurement on the file and move the file to the Inventory directory.

1. Click the Start or Start All button to begin the test:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



2. If the **Start** button is clicked, only the test highlighted in the list above will be started. If the **Start All** button is clicked, all tests in the list above will be started.

If the specified Degraded directory contains files when VQT testing is started, VQT will immediately begin processing the files. The quality tests will be performed and tabulated, then the files tested will be moved to the specified Inventory directory.

If the specified Degraded directory does not contain files when VQT testing is started, VQT will wait for a file to be placed in the directory. When a file is placed in the directory (usually by VQuad transfer), the VQT will automatically perform the measurement and move the file to the specified Inventory directory.

3. Click the **Stop All** button to end testing:

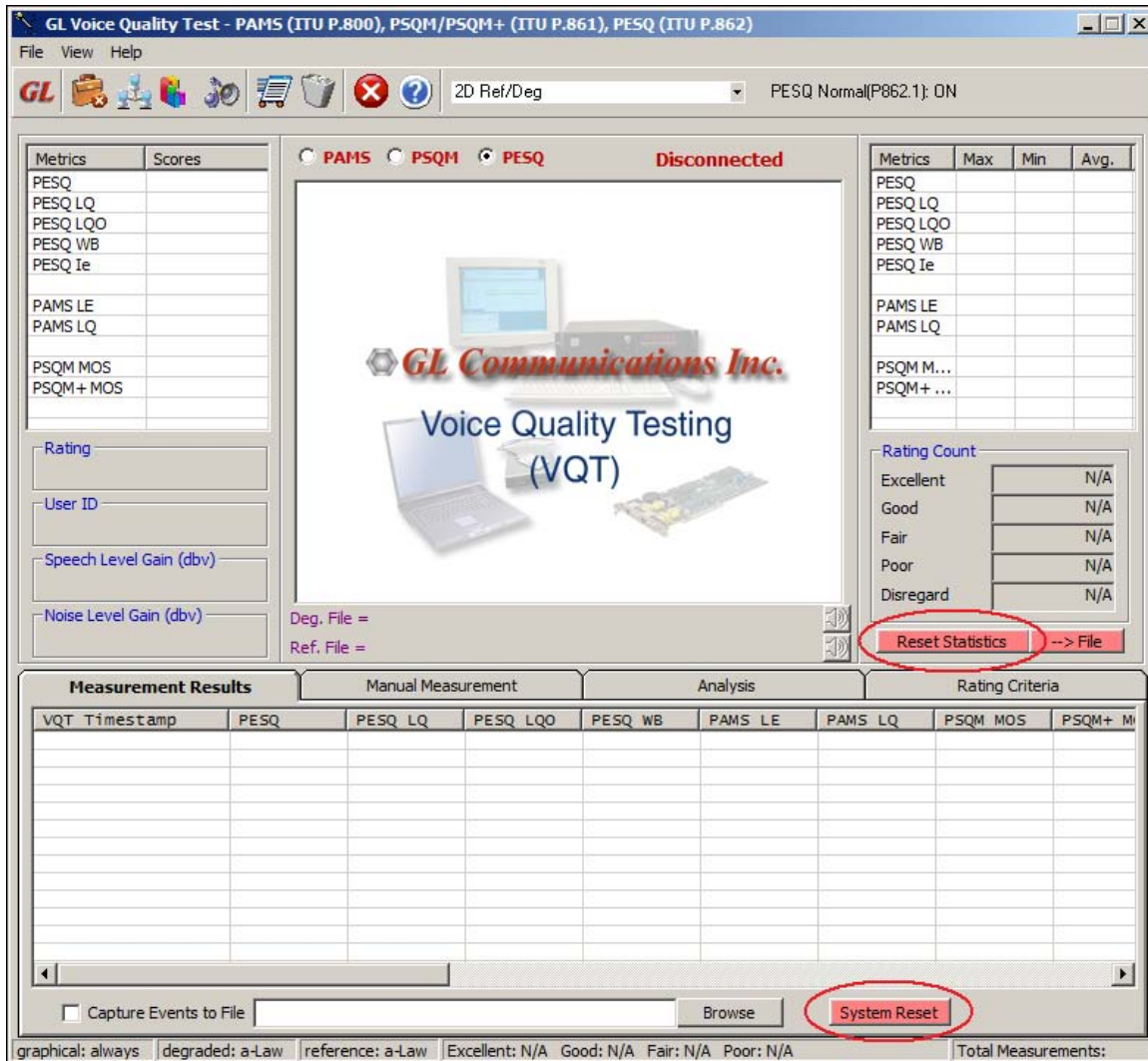


Before starting a test on transferred data, click the **Reset Statistics** and **Reset System** buttons to clear the display and measurement files:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



D-7 Test Execution

With the above configurations in place, the system is prepared to perform automated file transfer and voice file quality testing.

D-7.1 Preparation

1. Ensure that the **Reference** files specified in the **VQuad Auto-Traffic Tx Provisioning** and the **VQT Reference File** sections exist, and are of the correct (and matching) file format(s).
2. Ensure that the **VQT_Degraded** directory/directories specified in the **VQuad Auto-Traffic Tx Provisioning** section exist and are empty.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

3. Ensure that the **Inventory** directory/directories specified in the **VQT Inventory** sections exist and are empty.

D-7.2 Start VQT

1. Click the Start or Start All button to begin the test:



2. If the **Start** button is clicked, only the test highlighted in the list above will be started. If the **Start All** button is clicked, all tests in the list above will be started.

VQT is now in a 'Wait' state, and polling the **Degraded** directory or directories specified in the **VQT Setup**.

When a file is transferred into the **Degraded** directory by VQuad, VQT software will perform comparative testing with the degraded file and the file specified in the **VQT Reference** field of the **VQT Setup**. (Note: This should also be the file transferred by VQuad, as specified in the **Tx Provisioning** of **VQuad Auto-Traffic Configuration**.)

D-7.3 Start VQuad

In the VQuad Main Screen, click the Start All Traffic button:



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3



When files transfers are completed, click Stop All Traffic button from the VQuad™ Control Panel:



D-7.4 Transfer/Test Monitoring

VQuad Status Monitoring:

The VQuad call and transfer progress can be monitored in two primary locations on the



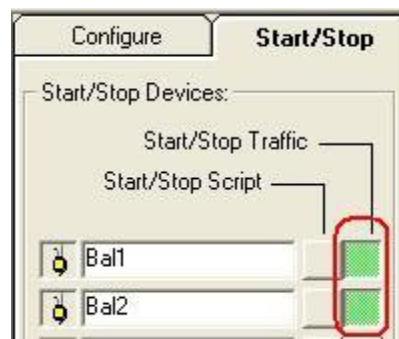
Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

VQuad Main Screen

1. Device Status

The upper area of the left panel of the **VQuad Main Screen** will highlight the blocks associated with configured devices when they are active:



2. Call Status

The lower area of the left panel of the **VQuad Main Screen** will indicate the status of calls in progress, and will indicate real-time transmission and receiving activity:

	Call Status	Traffic	Volume	Script
1.	Connected	Running		None
2.	Connected	Running		None

The **Call Status** section will also display activity of the individual devices:

Call Status Section Example 1

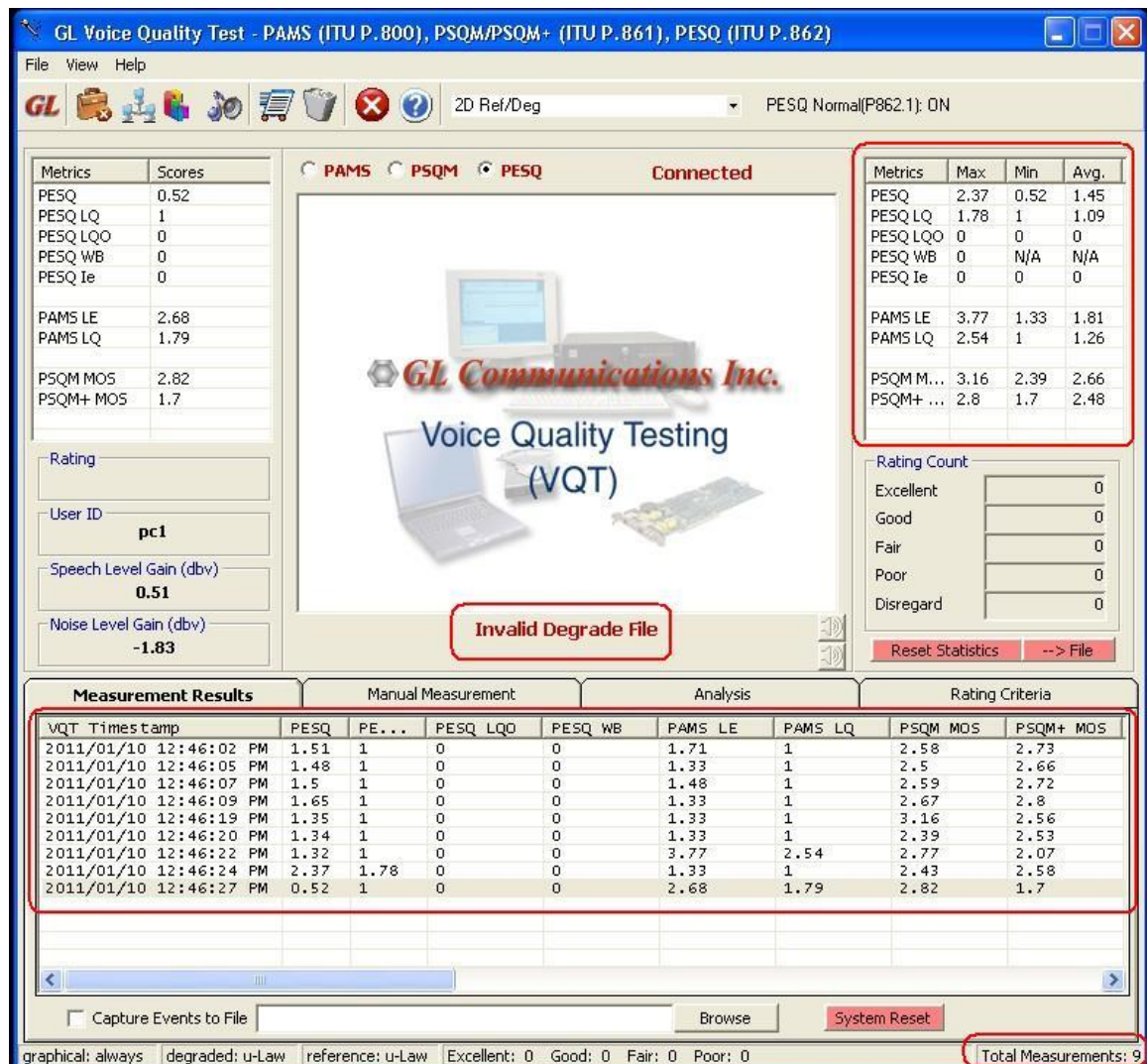
	Call Status	Traffic	Volume	Script
1.	Connected	1 Bx File		None
2.	Connected	1 Bx File		None

Call Status Section Example 2



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



D-7.6 File Movement Monitoring (Windows)

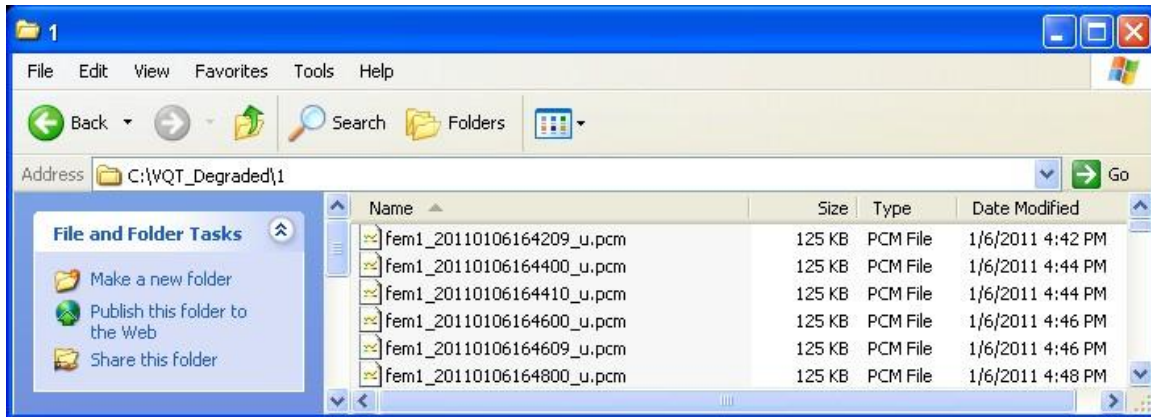
VQuad transfer progress can be monitored externally to VQuad by simply using Windows Explorer (or the Command Window) to view files as they are transferred into the **VQT_Degraded** directory:

VQT Degraded Directory (Explorer):

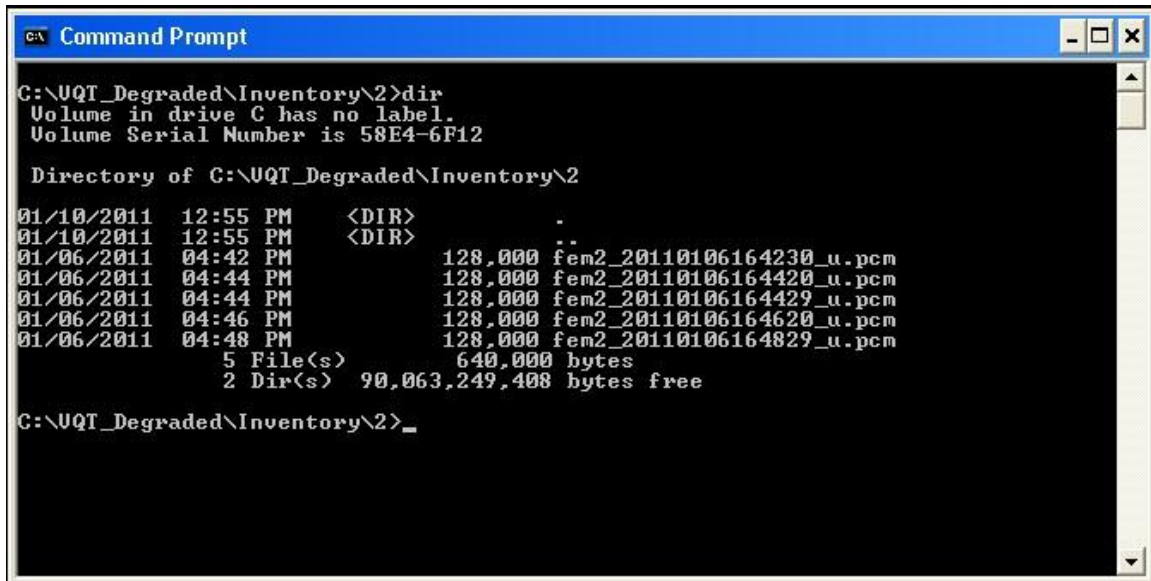


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



VQT Degraded Directory (Command Line):

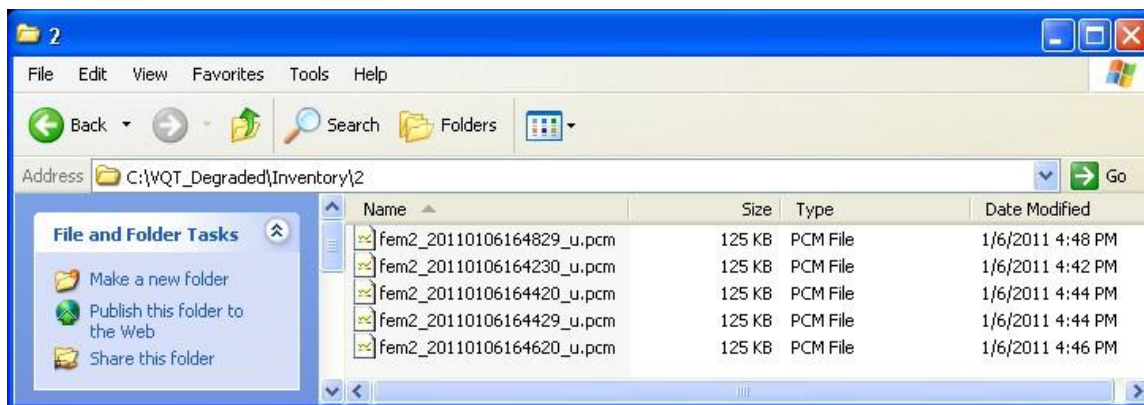


VQT progress can be monitored by Windows Explorer (or the Command Window) to view files as they are transferred from the **VQT_Degraded** directory into the **Inventory** directory:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



Note: If the transfer and test operations are running simultaneously, the time files remain in the **VQT_Degraded** directory may be very short; depending on circumstances, they may never become visibly listed. (This will occur when VQT testing moves the files from the **VQT_Degraded** before Windows Explorer has updated the listing.)

Progress can still be monitored, however, by observing the files entering the VQT **Inventory** directory.

D-8 Test Results

Test system hardware and software are integrated and implemented in accordance with Reference 3*.

The results of each run are tabulated and averaged for each Date/Time Group (DTG) of the run for the following data points:

1. PAMS_LE
2. PAMS_LQ
3. PSQM
4. PSQM_MOS
5. PSQMPlus
6. Calculated PSQMPlus_MOS
7. PESQ_Score
8. PESQ_LQ
9. PESQ_MOS

Detailed Data from Cisco Packetcable Implementation:

**GL Communications Voice Quality Testing (VQT) User Guide*, Updated June 2010



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The most familiar voice quality measurement is Mean Opinion Score (MOS). MOS is expressed on a five-point scale, with 5 being the best and 1 the worst. Table MOS-1 displays further detail about MOS ratings.

Table MOS-1

Rating	Speech Quality	Distortion Level
5	Excellent	Imperceptible
4	Good	Just perceptible, not annoying
3	Fair	Perceptible, slightly annoying
2	Poor	Annoying, but not objectionable
1	Unacceptable	Very annoying, objectionable

A MOS score of 4.0 is considered "toll quality," the quality associated with traditional PSTN service. Originally, MOS scores were purely subjective, based on individuals listening to voice samples and grading them. However, because external factors (including background noise at both ends of the communication path and even the individual's mood) could greatly influence scoring, more scientific ways of quantifying voice quality have been developed.

**** Note:** Historical research indicates that a 3 per cent packet loss results in a MOS score dropping by 0.5 (out of 5).

D-8.1 PAMS

The Perceptual Analysis/Masurement System (PAMS) was developed by British Telecom, and uses a mathematical model of human hearing. PAMS measurements report two scores: Listening Quality (LQ) and Listening Effort (LE).

These are both on a five-point scale similar to the MOS score.

**** Note:** Research shows that PAMS scores average within a half-point when compared to traditional MOS scores.

**** Note:** In PAMS measurements, errors in input signals are taken into consideration.

D-8.2 PSQM and PSQM+

The Perceptual Speech Quality Measure (PSQM) was developed by the ITU (the International Telecommunications Union) as standard P.861. It is designed to measure the effect of compression on voice quality; it does not take lost packets into account.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

PSQM scores are on a scale of 0 to 6.5, with 0 indicating no distortion and 6.5 indicating extreme distortion. (Thus, the lower the number, the better the voice quality.) There is no direct correlation between 'traditional' MOS scores and PSQM scores.

PSQM+ was developed to address some of these issues. PSQM+ performs additional post processing on PSQM values, and its results correlate more directly with 'traditional' MOS scores. It is to be noted, however, that PSQM+ still has issues with testing which involves lost packets.

D-8.3 PESQ

The Perceptual Evaluation of Speech Quality (PESQ) is presently the most advanced method for voice quality measurement. It was designed to succeed PSQM, and is defined in ITU standard P.862.

PESQ combines the techniques of PSQM+ and PAMS, and takes into account distortion, errors, packet loss, and delay.

PESQ measurement uses a sensory model (the comparison of the original ('reference') signal with the received ('degraded') signal), and its scores are analogous to MOS scores, with the best PESQ score which can be achieved being 4.5.

D-8.4 Test File Verification/Auditing

The testing system files for testing and analysis will be voice files in formats provided by GL Communications (as VQT_Reference).

The hardware, software, and files used will be verified and validated at unit and integrated levels prior to introduction of the newly developed elements. (See Appendix 1 for details of unit and integration verification and validation.)

In addition, each test will be performed with each configured User Agent (UA) in Master and in Slave mode.

The tests consist of three iterations ('runs') of each:

- 1 encoded voice file transferred 10 times
- 1 encoded voice file transferred 250 times
- 4 different encoded voice files transferred 10 times
- 4 encoded different voice files transferred 250 times
- 6 different encoded voice files transferred 10 times
- 6 different encoded voice files transferred 250 times

All transfers are bidirectional, with each DUT transmitting and receiving.



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

References

1. Department of Defense Unified Capabilities Requirements 2008 (UCR 2008)
2. Fusion *Voice Engine User's Manual*, Software Version 3.2.X, Part Number/Version: FVE V3.2.X, Release Date: November, 2010
3. *GL Communications Voice Quality Testing (VQT) User Guide*, Updated June 2010
4. *GL Communications VQuad™ (VOICE Quad) User Manual*, Updated May 2008
5. *GL Communications GL VQT Reference*, Log Output, September 2004
6. *Packetcable Implementation*, Jeff Riddel, Copyright 2007



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

**APPENDIX E Loopback Testing and Verification (Baseline
Verification)**



Table of Contents

E-1.	OVERVIEW AND ASSUMPTIONS	269
E-2.	OVERVIEW	269
E-3.	ASSUMPTIONS	269
E-4.	BASELINE VERIFICATION (LOOPBACK TESTING)	270
E-4.1	Detailed Implementation	270
E-4.1.1	Test File Verification/Auditing	270
E-4.1.2	Hardware Setup	270
E-4.1.3	Single-PC/Single-UTA Loopback Test Setup.....	271
E-4.1.4	Data Path.....	271
E-4.1.6	VQuad Configuration.....	271
E-4.1.7	Modify or Create a VQuad Device Configuration for Loopback Testing	274
E-4.1.8	Auto-Traffic Configuration	276
E-4.1.9	Configure Device 1	277
E-4.1.10	Configure Device 2.....	282
E-4.3.11	Modify or Create a VQuad Auto Traffic Configuration.....	285
E-4.3.12	General Tab Fields.....	286
E-4.3.13	Tx Provisioning Tab Fields	287
E-4.3.14	Rx Provisioning Tab Fields.....	287
E-4.3.15	VQT Setup	287
E-4.3.16	Checking VQT Auto-Measurement Setup.....	293
E-5	TEST EXECUTION	295
E-5.1	Start VQT.....	296
E-5.2	Start VQuad.....	296
E-6	TRANSFER/TEST MONITORING	297
E-6.1	VQuad Status Monitoring.....	297
E-6.2	VQT Testing Monitoring.....	298
E-7	TEST RESULTS.....	302
E-7.1	Detailed Data	303
E-7.2	PAMS.....	303
E-7.3	PSQM and PSQM+	304
E-8	UNIT AND INTEGRATION VERIFICATION AND VALIDATION	304



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

E-1. Overview and Assumptions

E-2. Overview

The purpose of this document is to detail the setup, implementation, and execution of testing voice quality in the use of the L-3 Maritime-developed SIP solution in progress.

The voice quality testing for the project will be done in three parts:

1. Baseline Verification ensures the functionality of individual components and test elements, and includes verifying:
 - Equipment/setup
 - Software/setup
 - Test files
2. RTP/Transfer Testing evaluates the performance of the Real-Time Protocol (RTP) to be implemented in the integrated solution. It consists of:
 - Establishing communications links over RTP
 - Transferring selected voice files over the link
 - Evaluating the degradation of files transferred
3. End-to-End Communication Verification tests the VoIP/AS-SIP stack (for call control) and the RTP (for data/voice transfer), and the selected endpoint nodes as an integrated system, and consists of:
 - Call completion and traffic handling analysis and evaluation
 - Voice file transfer and analysis and evaluation

E-3. Assumptions

- Voice files will be transferred and analyzed using u-Law encoding as the standard.
- GL Communications VQT software will be used to evaluate the integrity and parameters of transferred voice files.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

E-4. Baseline Verification (Loopback Testing)

E-4.1 Detailed Implementation

The baseline for the testing system is defined as the hardware, software, configuration, and files used for testing and verification of the VoIP/AS-SIP solution. Following are the Details of the present and planned elements of the baseline.

E.4.1.1 Test File Verification/Auditing

The testing system files for testing and analysis will be voice files in formats provided by GL Communications (as VQT_Reference).

The hardware, software, and files used will be verified and validated at unit and integrated levels prior to introduction of the newly developed elements. (See Appendix 1 for details of unit and integration verification and validation.)

E-4.1.2 Hardware Setup

The testing system hardware consists of:

1. Two equivalent Dell Latitude D630 laptops as processing nodes
2. Two GL Communications Universal Telephony Agents (UTAs)
3. Connectors and cabling necessary for interconnection

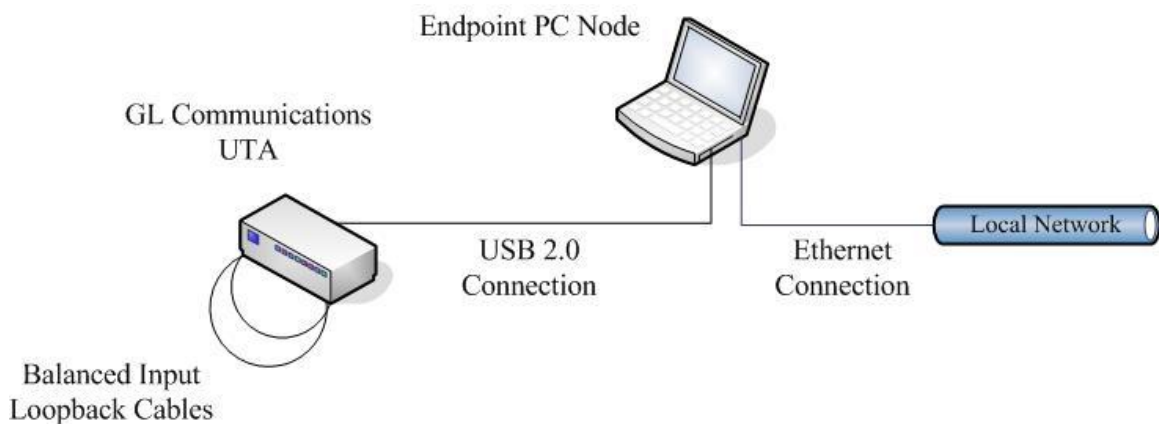


Figure 0-1 Loopback Transfer Test Setup

Initial tests will consist of a single PC and a single UTA. For verification purposes, tests will be run with all PC/UTA combinations (see ‘Test Results’ for test tabulation summary.)



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

E-4.1.3 Single-PC/Single-UTA Loopback Test Setup

The configuration used for Loopback testing is a one-node system on a single PC with a Dual UTA (Audio IN to Audio OUT).

On the Dual UTA, cross-connect using 3.5mm mono audio cables:

1. Audio IN of Side #1 to the Audio OUT of Side #2
2. Audio OUT of Side #1 to the Audio IN of Side #2

Ensure the UTA is connected to a PC running VQuad™ via a USB 2.0 connection.

E-4.1.4 Data Path

Selected voice files will be transferred from the Endpoint PC Node to the UTA, which will perform an internal loopback transfer using the Balanced inputs and outputs. The transferred files will then be sent to the Endpoint PC Node and stored in the local VQT_Degraded directory. (See Software Setup\VQuad Configuration.)

E-4.1.5 Software Setup

The testing and analysis system software consists of:

1. Windows XP SP3 (operating system for processing nodes)
2. GL Communications VQuad (file transfer and traffic generation software)
3. GL Communications VQT (Voice Quality Testing and analysis software)
4. Wireshark and other utilities necessary for support and analysis
5. Microsoft Excel and Word for data analysis and presentation

E-4.1.6 VQuad Configuration

To configure GL Communications VQuad for Loopback Testing:

1. Start VQT by double-clicking the GL VQuad icon:

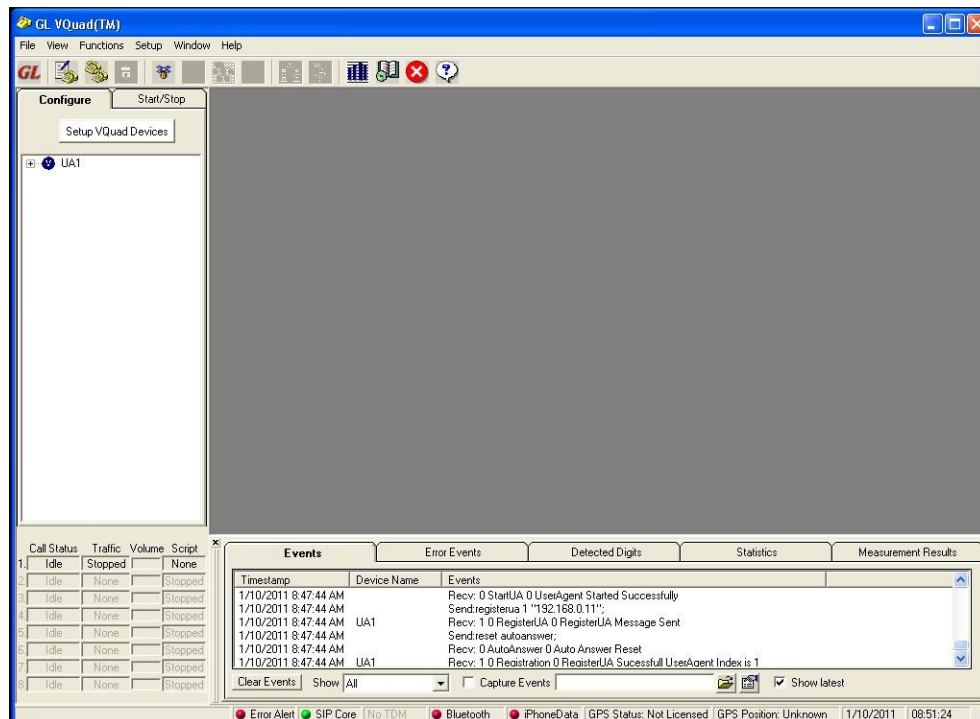


The VQuad Main Screen appears.

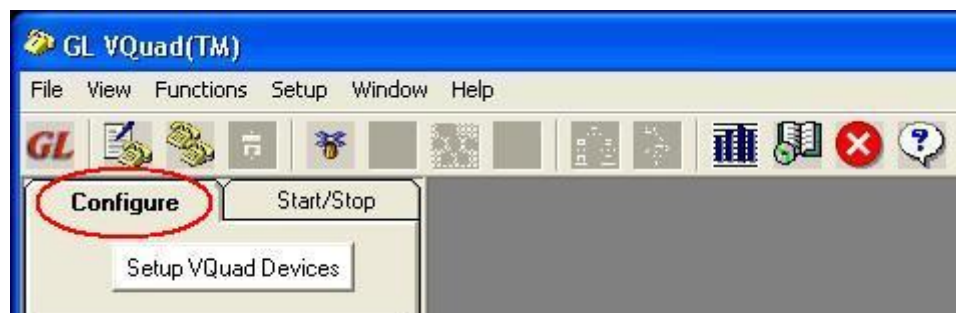


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



2. Click the Configure tab:



3. A custom device configuration has been created for this test implementation. To access and use the configuration, click the Setup VQuad Devices button:





Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- Click the Device Configuration – Load button:



- Select the configuration PCx_Loopback_20x1.

The naming scheme for device configurations is as follows:

For the configuration PCx_Loopback_20x1,

PCx_ indicates the PC testing node for which the configuration is designed:

- PC1 indicates Endpoint PC Node 1
- PC2 indicates Endpoint PC Node 2
- PCx indicates the configuration can be used for either Endpoint PC Node

Loopback_ indicates the type of test:

- Loopback indicates the test is single-node, endpoint-to-same-endpoint
- Transfer indicates the test is multi-node (two or more nodes), transfer endpoints different
- EndToEnd indicates that all devices in a VoIP/AS-SIP communication path will be tested

20x1 indicates the number of transfers and the number of files:

- The first number (in this case, 20) is the total number of transfers to be performed
- The second number (in this case, 1) is the number of files which will be transferred

Thus, the configuration name PC2_Transfer_100x4 would indicate:

- The configuration is designed for PC2
- It is a multi-node transfer configuration
- 100 transfers of four files will be performed

- Click the Exit button to save the configuration selection and exit.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

E-4.1.7 Modify or Create a VQuad Device Configuration for Loopback Testing

If the required configuration is not available, or another configuration is to be created, use the following procedure:

1. Click the Device Configure tab:



2. Click the Setup VQuad Devices button:



For loopback testing, two devices will be set up as 'Balanced Input'.

3. In the GL VQuad™ Device Configuration window, click the Number of Devices down arrow and select 2.



4. Click the Device Type down arrow and select Dual UTA as the device type.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

GL VQuad(TM) Device Configuration

Number of Devices
Number: 2

Device Configuration
Load Save Delete

	Device Type	Device Name	UTA Type
1.	Dual UTA	Bal1	Balanced
2.	Dual UTA	Bal2	Balanced

Note: Exit VQuad(TM) Software Prior to unPlug Dual UTA

5. Enter a name in the Device Name field. For illustration purposes, the Device Names entered will be Bal1 and Bal2:

GL VQuad(TM) Device Configuration

Number of Devices
Number: 2

Device Configuration
Load Save Delete

	Device Type	Device Name	UTA Type
1.	Dual UTA	Bal1	Balanced
2.	Dual UTA	Bal2	Balanced

Note: Exit VQuad(TM) Software Prior to unPlug Dual UTA

6. Select UTA Type as Balanced I/O for both ports. (Dual UTA port numbers are automatically displayed with the loopback connection):

GL VQuad(TM) Device Configuration

Number of Devices
Number: 2

Device Configuration
Load Save Delete

	Device Type	Device Name	UTA Type
1.	Dual UTA	Bal1	Balanced
2.	Dual UTA	Bal2	Balanced

Note: Exit VQuad(TM) Software Prior to unPlug Dual UTA



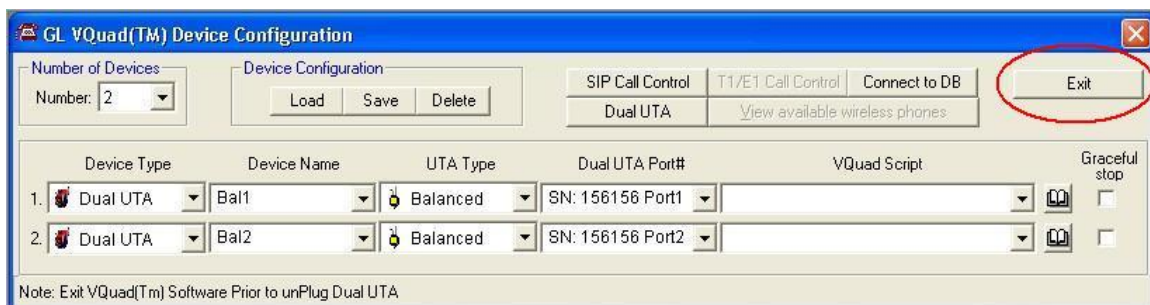
Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

7. Save the device configuration with a descriptive name using the Save button. For illustration purposes, the configuration will be saved as Balanced_Input_1:



8. Click the Exit to save the configuration and exit Device Configuration:



VQuad devices are now configured to perform loopback transfers using the two balanced inputs on the attached UTA.

E-4.1.8 Auto-Traffic Configuration

Auto-Traffic Configuration determines the method of transfer (master/slave identification), the type of file transferred, and the locations of files transmitted and received.

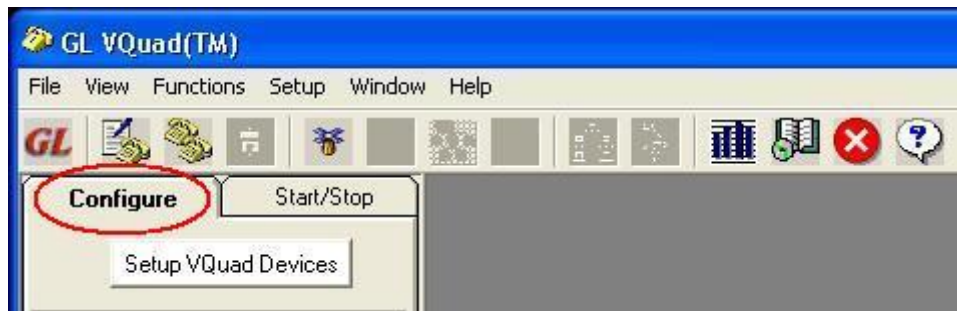
Custom configurations have been created for this test implementation. To access and use the configurations, use the following procedure:

In the VQuad Main Window, click the Configure tab:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



E-4.1.9 Configure Device 1

1. Double-click Bal1 (or any other name configured to Device 1 in the previous procedure) in the Devices panel on the left of the VQuad Main Window. Alternately, click the + sign to the left of the named device:



The Bal1 detail categories will be listed:



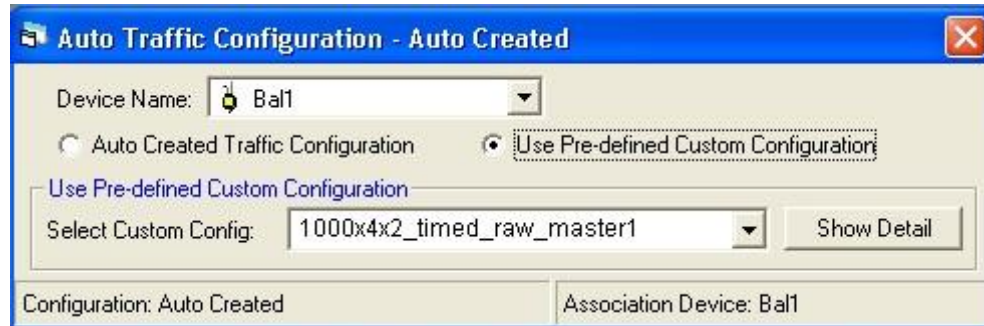


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

2. Double-click the Auto-Traffic entry.

The Auto-Traffic Setup window will appear:



3. Select Bal1 as the Device Name:



4. Click the Use Pre-defined Custom Configuration radio button:



5. Click the down arrow to the right of the Configuration Name field to access the list of available configurations.
6. Select the configuration Loopback_20x1_Master for Device 1.

The naming scheme for Auto-Traffic configurations is as follows:

For the configuration Loopback_20x1_Master,



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Loopback indicates the type of test:

- Loopback indicates the test is single-node, endpoint-to-same-endpoint
- Transfer indicates the test is multi-node (two or more nodes), transfer endpoints different
- EndToEnd indicates that all devices in a VoIP/AS-SIP communication path will be tested

20x1 indicates the number of transfers and the number of files:

- The first number (in this case, 20) is the total number of transfers to be performed
- The second number (in this case, 1) is the number of files which will be transferred

Master indicates the selected device's role in the transfers:

- Master: In this mode, the VQuad™ will be in master configuration and transmits the reference files at a regular time interval (set by the operator) regardless of whether or not it receives any incoming degraded files from the other device.
- Slave: This mode transmits a VQuad™ reference file only in response to receiving an incoming degraded file.

Thus, the configuration name PC2_Transfer_100x4 would indicate:

- The configuration is designed for PC2
- It is a multi-node transfer
- 100 transfers of four files will be performed

After selecting the configuration Loopback_20x1_Master, ensure that the following parameters are correct for the relevant device:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'Auto Traffic Configuration - Loopback_20x1_Master' window with the 'Auto Trigger' tab selected. The 'Master' sub-tab is active. The 'Trigger On Time' section is expanded, showing the following settings: Cycle Time (s) is 30, Loop Period (s) is 120, Stop Iterations is 20, File Length (s) is 8, Tx/Rx Offset (s) is 6, and Send Delay (ms) is 1500. The 'Trigger On DTMF digits', 'Trigger On MF digits', and 'Trigger On User Defined Tones' options are not selected. At the bottom, there are buttons for 'Load Configuration' and 'Save As Configuration', and a status bar showing 'Configuration: Loopback_20x1_Master' and 'Association Device: Bal1'.

Loopback_20x1_Master Auto-Trigger Tab

The screenshot shows the 'Auto Traffic Configuration - Loopback_20x1_Master' window with the 'General' tab selected. The 'File Format' section shows '8000; 16-bit; PCM' selected. The 'Increment Rx File Name' section shows 'Time Stamp' selected. The 'Digit Parameters' section shows 'Enable Digit Tx' and 'Enable Multiple Digit Detection' both unchecked. The 'On Time (ms)' is 200, 'Off Time (ms)' is 750, 'High Power (-dB)' is 10, and 'Low Power (-dB)' is 10. At the bottom, there are buttons for 'Load Configuration' and 'Save As Configuration', and a status bar showing 'Configuration: Loopback_20x1_Master' and 'Association Device: Bal1'.

Loopback_20x1_Master General Tab

[illegible]

Loopback_20x1_Master Tx Provisioning Tab

Auto Traffic Configuration - Loopback_20x1_Master

Auto Trigger General Tx Provisioning **Rx Provisioning**

Timing	Degraded File Path	Time (ms)
8	C:\WQT_Degraded\1\mem1	7000
38		7000
68		7000
98		7000
		7000
		7000
		0
		0
		0
		0

Configuration: Loopback_20x1_Master Association Device: Ball

Loopback_20x1_Master Rx Provisioning Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

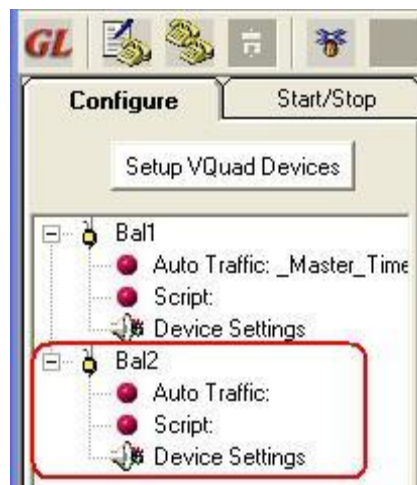
Maritime Systems

E-4.1.10 Configure Device 2

1. Double-click Bal2 (or any other name configured to Device 1 in the previous procedure) in the Devices panel on the left of the VQuad Main Window. Alternately, click the + sign to the left of the named device:

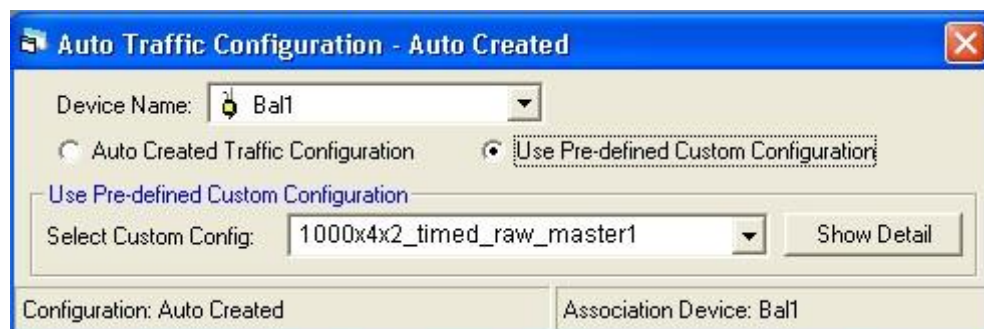


The Bal2 detail categories will be listed:



2. Double-click the Auto-Traffic entry.

The Auto-Traffic Setup window will appear:



3. Select Bal2 as the Device Name:

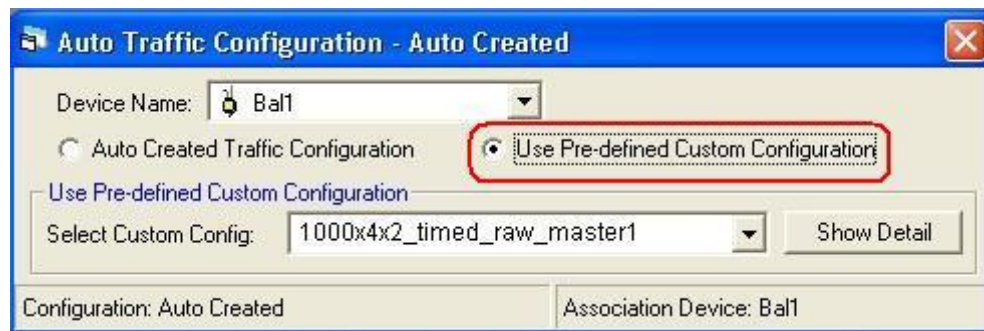


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

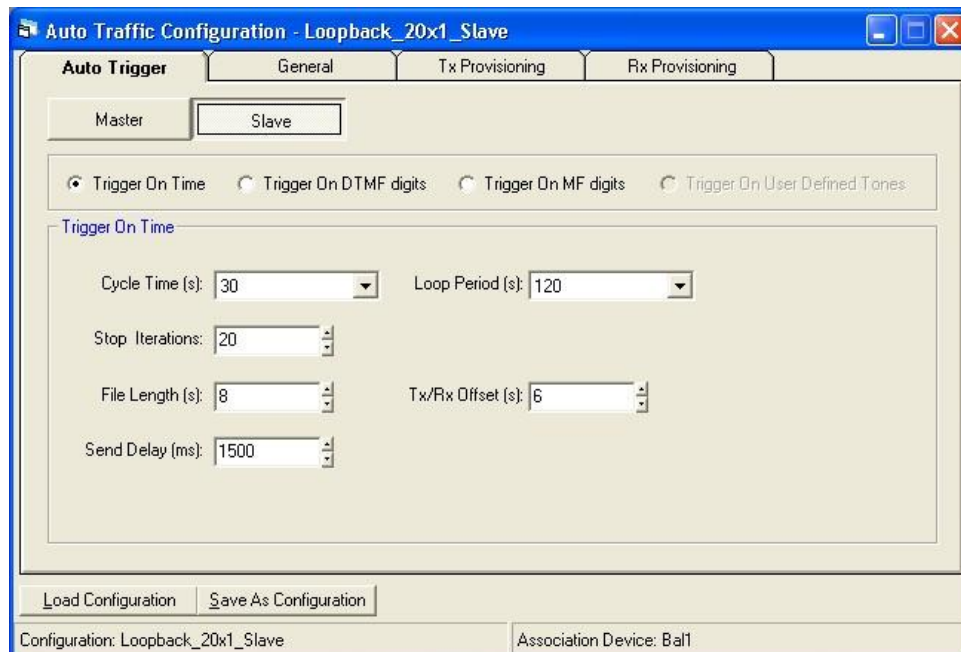
Maritime Systems

Replace_with_VQuad_AutoTraffic_ConfigurationButton_Bal2DeviceNameIndicated.JPG

- Click the Use Pre-defined Custom Configuration radio button:



- Click the down arrow to the right of the Configuration Name field to access the list of available configurations.
- Select the configuration Loopback_20x1_Slave for Device 2.
- After selecting the configuration Loopback_20x1_Slave for Device 2, ensure that the following parameters are correct for the relevant device:



Loopback_20x1_Slave Auto-Trigger Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'Auto Traffic Configuration - Loopback_20x1_Slave' window with the 'General' tab selected. The window has four tabs: 'Auto Trigger', 'General', 'Tx Provisioning', and 'Rx Provisioning'. The 'General' tab contains the following settings:

- File Format:** Three radio buttons are present: '8000; 8-bit; A-Law', '8000; 8-bit; Mu-Law', and '8000; 16-bit; PCM'. The '8000; 16-bit; PCM' option is selected.
- Increment Rx File Name:** Three radio buttons are present: 'Sequential', 'Time Stamp', and 'GPS + Time Stamp / ITS + Time Stamp'. The 'Time Stamp' option is selected.
- Digit Parameters:** Two checkboxes are present: 'Enable Digit Tx' and 'Enable Multiple Digit Detection'. Both are unchecked. To the right of these are four spin boxes: 'On Time (ms)' set to 200, 'High Power (-dB)' set to 10, 'Off Time (ms)' set to 750, and 'Low Power (-dB)' set to 10.

At the bottom of the window, there are two buttons: 'Load Configuration' and 'Save As Configuration'. Below these buttons, the text 'Configuration: Loopback_20x1_Slave' and 'Association Device: Bal1' is displayed.

Loopback_20x1_Slave General Tab

The screenshot shows the 'Auto Traffic Configuration - Loopback_20x1_Slave' window with the 'Tx Provisioning' tab selected. The window has four tabs: 'Auto Trigger', 'General', 'Tx Provisioning', and 'Rx Provisioning'. The 'Tx Provisioning' tab contains the following settings:

- Timing:** A vertical list of four radio buttons: '0', '30', '60', and '90'. The '0' option is selected.
- Reference File Path:** A text box containing the path 'C:\WQT_Reference\WQuad_Auto\Raw\fm1_1.pcm'. To the right of the text box are eight file selection icons.
- Hold Off Time (ms):** A spin box set to 2000.
- Identifier Digit:** Two radio buttons: 'In File' and 'Generate'. The 'Generate' option is selected.
- Power (dB):** A spin box set to 0.
- Duration (ms):** A spin box set to 0.

At the bottom of the window, there are two buttons: 'Load Configuration' and 'Save As Configuration'. Below these buttons, the text 'Configuration: Loopback_20x1_Slave' and 'Association Device: Bal1' is displayed.

Loopback_20x1_Slave Tx Provisioning Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Timing	Degraded File Path	Time (ms)
8	C:\WQT_Degraded\1\mem1	7000
38		7000
68		7000
98		7000
		7000
		7000
		0
		0
		0
		0

Loopback_20x1_Slave Rx Provisioning Tab

E-4.3.11 Modify or Create a VQuad Auto Traffic Configuration

If the required configuration is not available, use the following procedure to modify it or create a new one:

Configure the Auto Trigger, General, Tx Provisioning, and Rx Provisioning tabs in accordance with applicable requirements.

Guidelines for configuring the fields of the tabs follow:

Auto Trigger Tab Fields:

- **Cycle Time:** This is the time allotted for the VQuad™ master and slave operations to take place. This time should be adequately long when bidirectional operations are performed.
- **Loop Period:** This is the total length of a timed cycle. (The default profile uses six files in a cycle.)
- **Stop Iteration:** Specifies the number of iterations (cycles multiplied by number of lines configured) that the VQuad™ will execute before stopping the test.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- **File Length:** This indicates the recording duration and is required to set up the VQuad™ time triggering flow. This includes the time to receive an entire file length plus the send delay time.
- **Tx/Rx Offset:** This is the ‘wait period’ after completion of a Tx action, but before the return of Tx action for bi-directional file transfer.
- **Send Delay:** This is the time after which the VQuad™ receiving end starts recording. This is a buffer time which helps counter the clock variations of two independent PCs.

E-4.3.12 General Tab Fields

- **File Format:** This setting selects the format of the file that the VQuad™ receiver will be saved to after the receiver completes recording of an incoming degraded file. The recorded degraded file type must match the type of VQuad™ reference file being sent, or the VQT Analysis program will yield very bad VQT scores due to mismatched file types. VQuad™ reference files are provided in linear PCM, A-law, and Mu-law encoding formats. There are usually small differences in VQT scores when using one file type as opposed to another. VQuad™ normally executes using linear PCM Reference files.
- **Note:** When performing transfers using Dual UTAs, the file type must be 16-bit 8000 Hz sampled, Little Endian (Intel) PCM.
- **Increment Rx File Name:** When the VQuad™ completes recording of an incoming degraded voice file, the file several options can be used to suffix the received file name to simplify identifying the degraded files.

The filename prefix for each recorded sample is defined by the filename rules setup in the Rx Provisioning tab.

- **Sequential:** The sequential option simply appends a sequential number to each sample filename in the order as they are received. By default, the numbering starts from zero, but the user can select Sequential File Numbering to start the numbering sequence at any number, or to reset a sequence that was previously started.
- **Time Stamp:** Each incoming/recorded-degraded file has the time stamp (from the PC clock) that the sample was received as the filename suffix. (Note: Ensure that the PC clock is set to the correct time.)
- **Digit Parameters**



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- The Digit/Tone Parameters only work in conjunction with the User-defined Tone Triggering method when trigger tones external to the sample voice file are required.
- If Enable Digit TX is not checked, the VQuad™ program uses MF, DTMF or User-defined tones, attached (prefixed) to the VQT Reference File, to trigger the recording and to identify the VQT Reference File that follows. If Enable Digit TX is checked, the VQuad™ program sends tones according to the table of user-defined tones before sending voice file. In this case, the reference voice files without prefixed tones must be used. (Refer to section Tone/Digit Method for Sending/Recording for details)
- Enable Multiple Digit Detection is used only in ‘master mode’ with DTMF/MF trigger. It allows multiple digits detection during one session.
- For example, in normal case (Enable Multiple Digit Detection is not checked), one session consists of a ‘TX file’ and a ‘RX file’. Even if it detects another digit, it will wait till the end of the cycle duration and then proceeds to the next session. However, the Enable Multiple Digit Detection option when checked, will again allow “RX” file event to occur after another digit is detected.
- Enter On Time (200ms or longer), Off Time (750ms or longer), High Power (-dB) and Low Power (-dB) values.

E-4.3.13 Tx Provisioning Tab Fields

- Reference File Path: This field determines the location of the file(s) to be transferred.

Sample files are provided by GL Communications, and are located in local subdirectories of C:\VQT_Reference\.

E-4.3.14 Rx Provisioning Tab Fields

- Degraded File Path: The path on the local PC which is the destination for files transferred in the assigned time slot.

The final characters of the Degraded File Path entry will become the initial characters of the files placed in the Degraded File Path directory. These characters will be suffixed with the Sequential or Time Stamp data (configured in the General tab).

E-4.3.15 VQT Setup

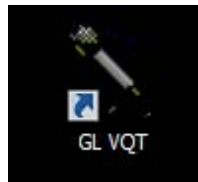
To configure GL Communications VQT for Auto Measurement Setup and Execution:

1. Start VQT by double-clicking the GL Voice Quality Testing icon:

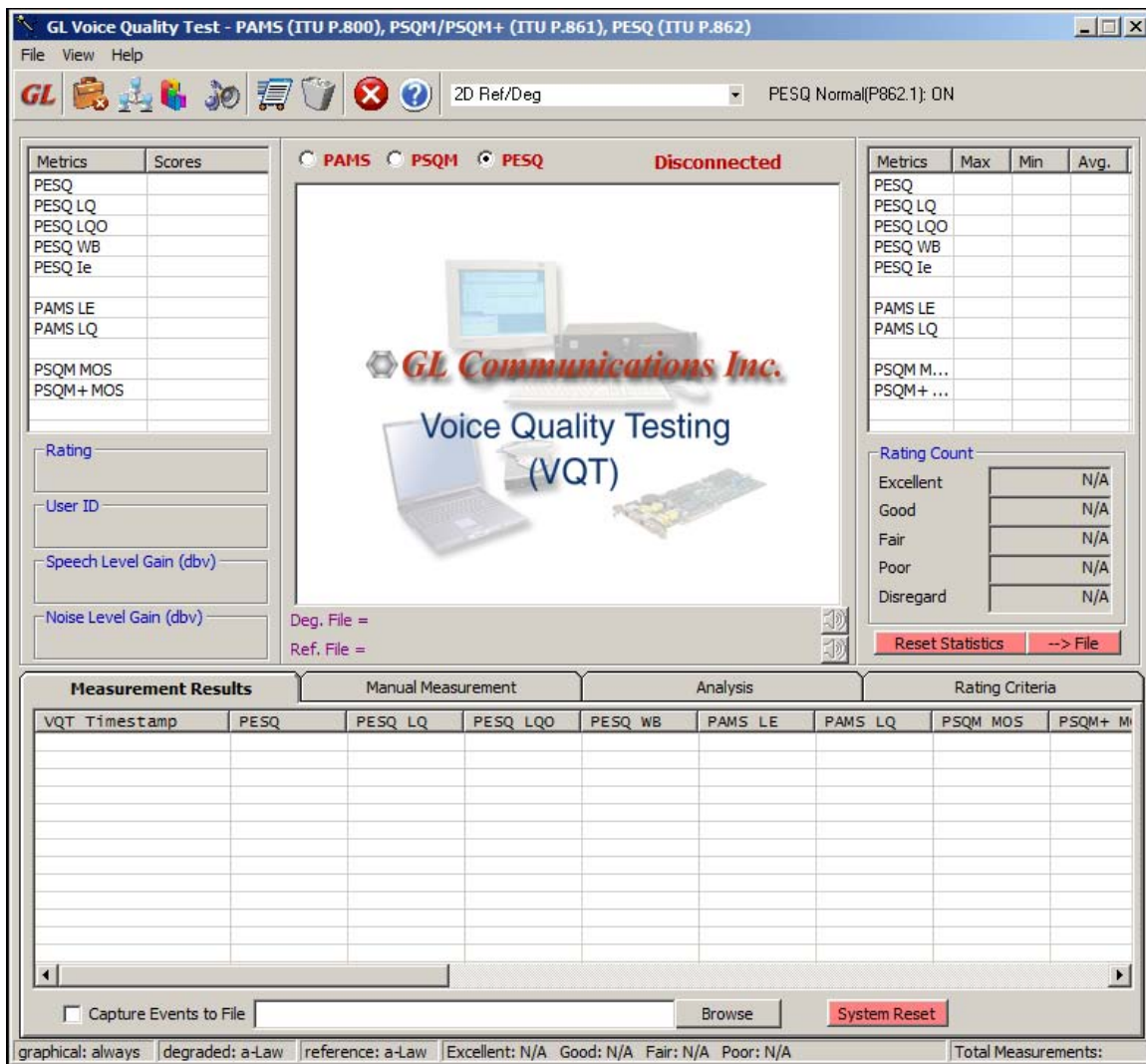


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



2. The VQT Main Screen appears:

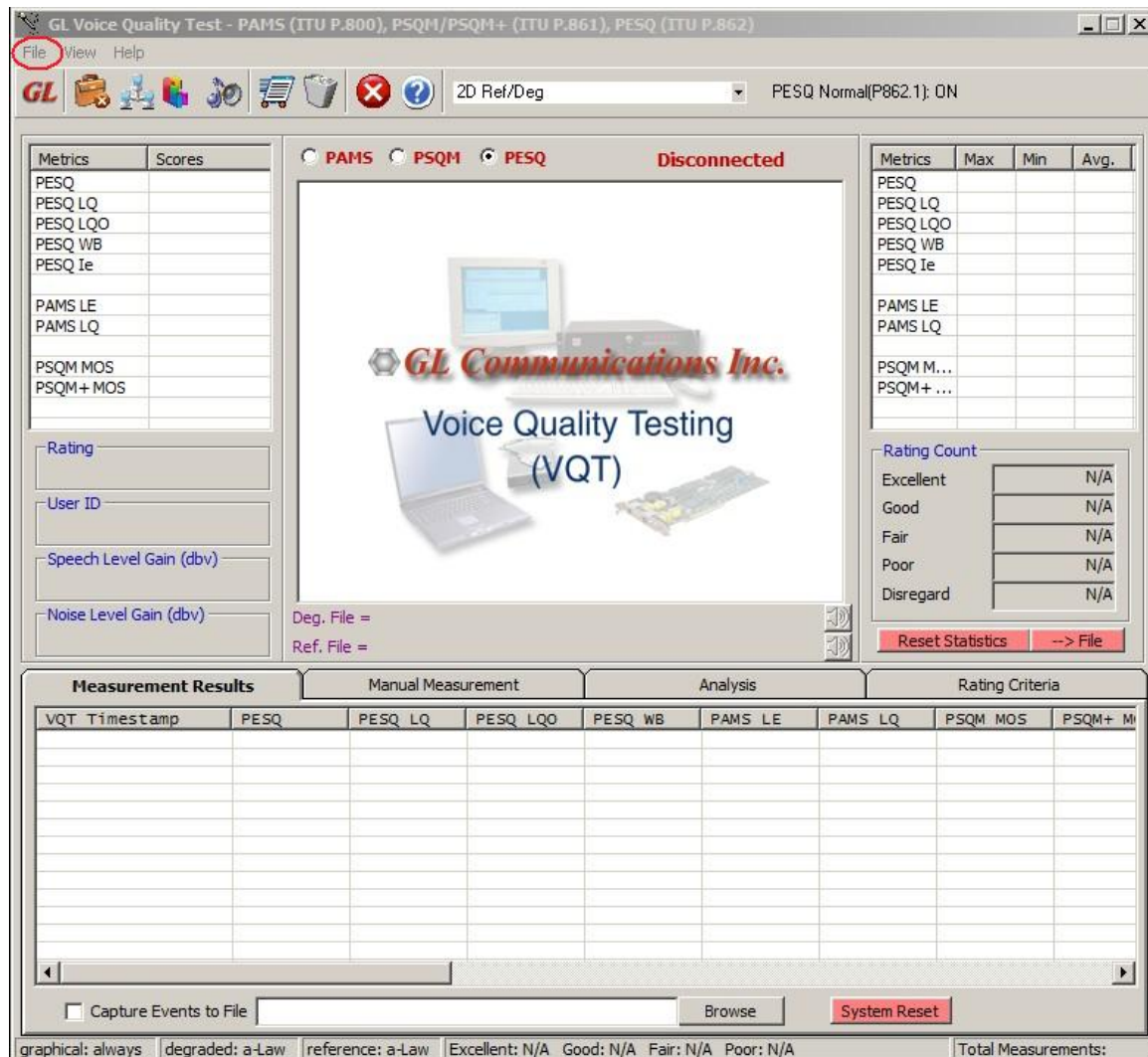


3. From the VQT main screen, select File -> Auto Measurement:

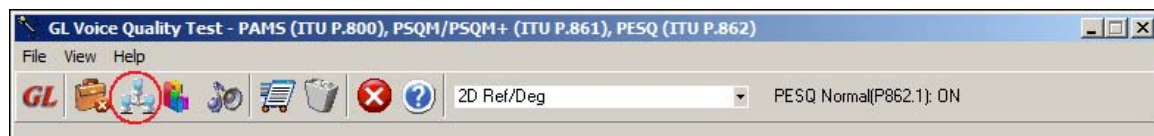


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



Alternatively, the Auto-Measurement icon can be clicked:



4. The VQT Auto-Measurement screen appears:

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

[illegible]

5. Click the Add button:

[illegible]

6. Populate the indicated fields to set up VQT Auto-Measurement:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

7. Enter the Degraded Directory in the space provided. This is the directory containing the files transferred in the VQuad Loopback Test, and is configured in VQuad Auto Traffic Configuration. (See VQuad Auto Traffic Configuration – Rx Provisioning Tab.)

For illustration purposes, C:\VQT_Degraded\1 will be entered.

Note: Ensure that C:\VQT_Degraded\1 (or whatever directory is specified as the Degraded Directory) exists on the hard drive. Clicking the Open Folder icon will open a Windows selection dialogue which will allow selection from existing directories:

8. Enter the Reference File in the space provided. This is the original file which was transferred according the VQuad Loopback Test, and is configured in VQuad Auto Traffic Configuration. (See VQuad Auto Traffic Configuration – Tx Provisioning Tab.)

For illustration purposes, C:\VQT_Reference\VQuad_Auto\fem1.pcm will be selected.

9. Select the Save degraded files to an inventory directory after measurement radio button.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

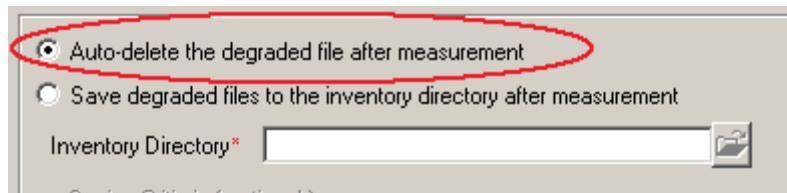
10. Enter the Inventory Directory in the space provided. This is the directory to which files will be transferred after testing. Thus, the file path is:

Degraded directory (specified in Step a) above)

For illustration purposes C:\VQT_Degraded\Inventory\1\ will be selected.

Note: Ensure that C:\VQT_Degraded\Inventory\1 (or whatever directory is specified as the Degraded Directory) exists on the hard drive. Clicking the Open Folder icon will open a Windows selection dialogue which will allow selection from existing directories.

11. Alternately, the Auto-delete the degraded file after measurement radio button can be selected:



If this button is selected, the transferred files will be deleted from the Degraded directory after testing. This option may be preferred if the test parameters are known correct, and the test measurements will not be repeated.

However, for purposes of data retention, or if running any given tests again is necessary or possible, the use of the Save degraded files to an inventory directory after measurement is preferable.

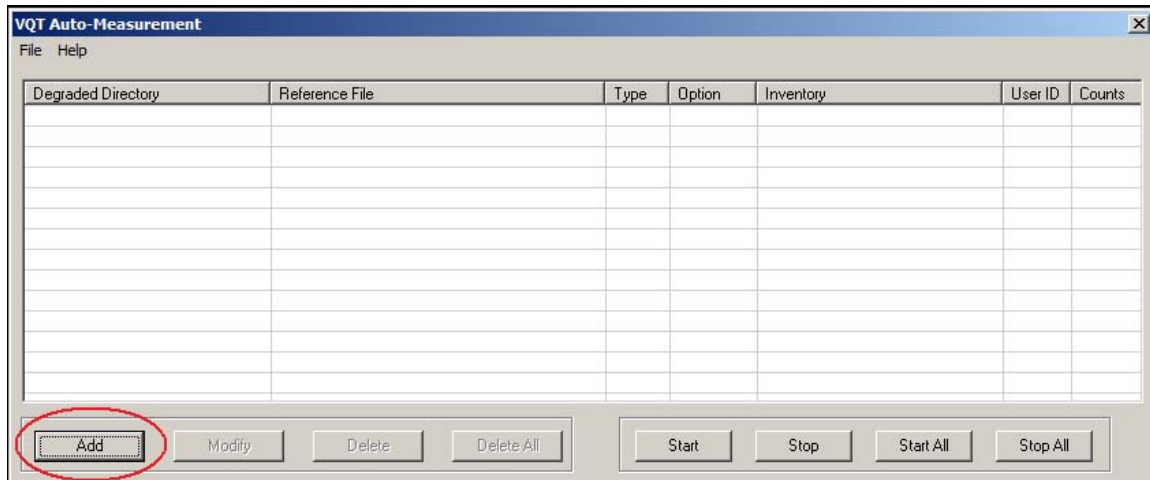
12. Within the User ID field, enter a User ID for tracking purposes. This field is not verified or cross-checked; it is stored as part of the data produced by the test, and can be useful for granular data analysis in a multi-testing environment.

13. Select the Add button to add the session:

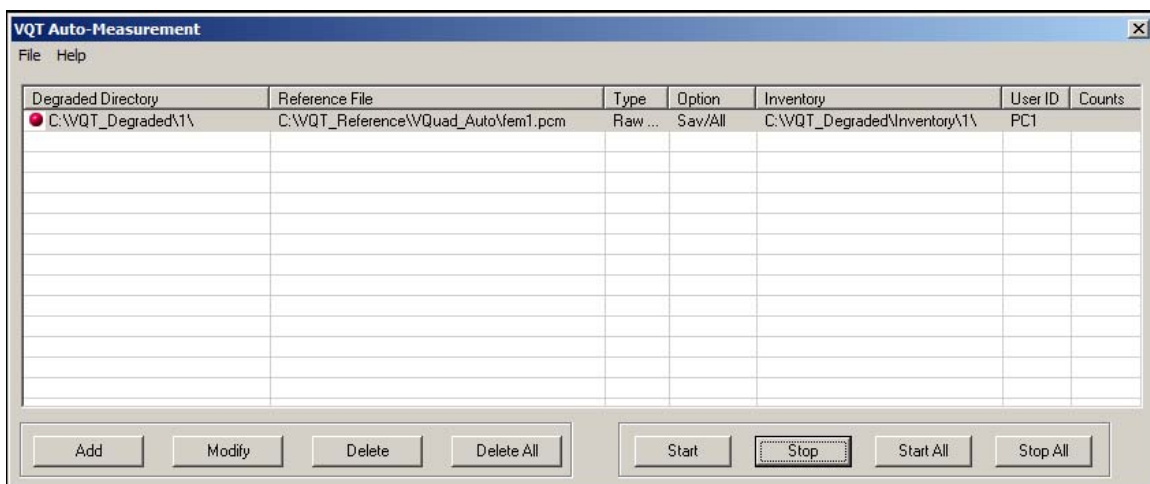


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



14. The VQT Auto-Measurement window will now indicate the data associated with the test entered:



E-4.3.16 Checking VQT Auto-Measurement Setup

To verify that the VQT setup is functional, manually copy a file of the correct format into the specified Degraded directory. If setup is correct, upon test execution VQT will automatically run the measurement on the file and move the file to the Inventory directory.

1. Click the Start or Start All button to begin the test:



2. If the Start button is clicked, only the test highlighted in the list above will be started. If the Start All button is clicked, all tests in the list above will be started.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

If the specified Degraded directory contains one or more files when VQT testing is started, VQT will immediately begin processing the files. The quality tests will be performed and tabulated, then the files tested will be moved to the specified Inventory directory.

If the specified Degraded directory does not contain files when VQT testing is started, VQT will wait for a file to be placed in the directory. When a file is placed in the directory (usually by VQuad transfer), VQT will automatically perform the measurement and move the file to the specified Inventory directory.

3. When the tests have been verified to run correctly, click the Stop All button to end testing:

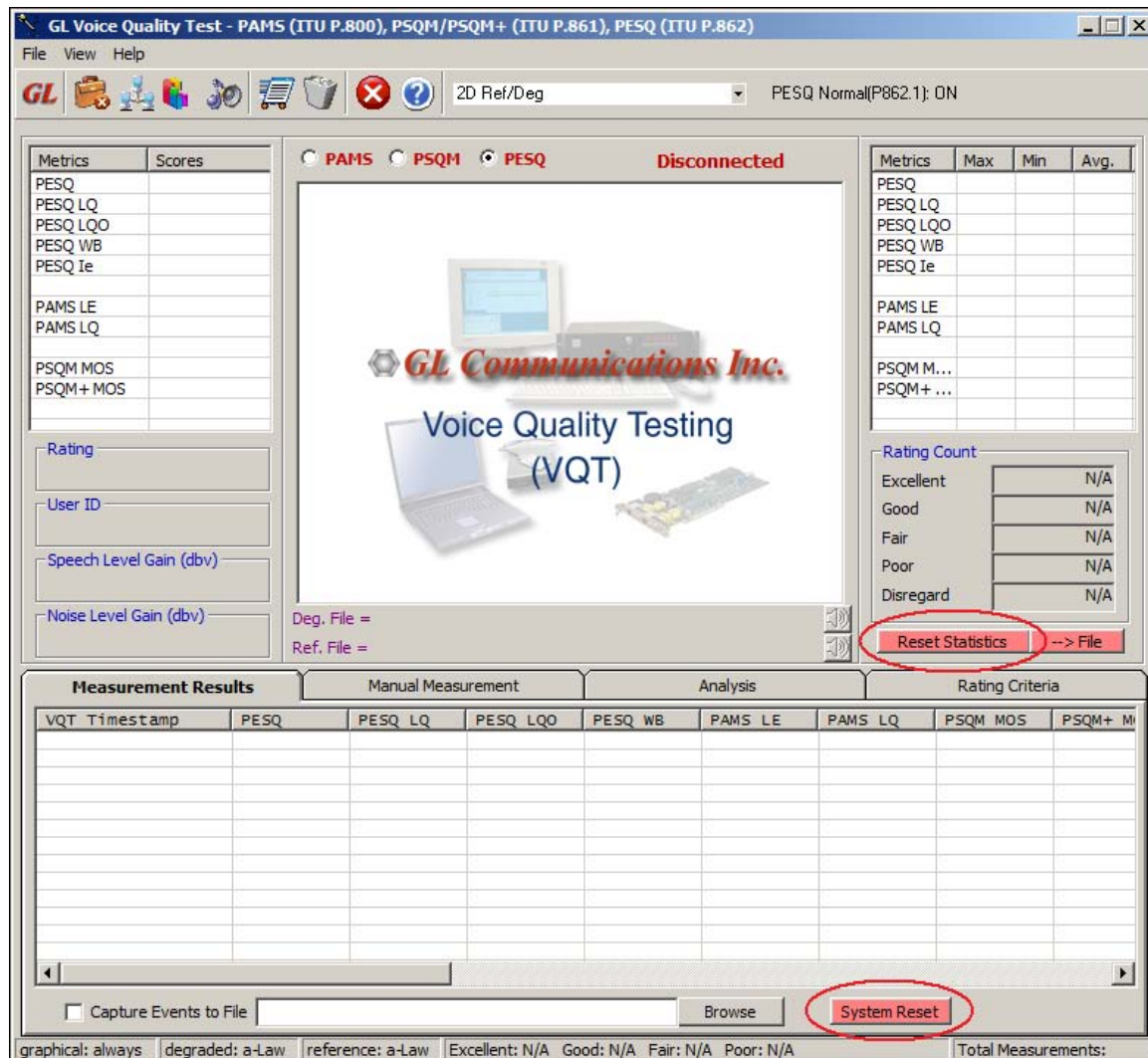


Before starting a test on transferred data, click the Reset Statistics and Reset System buttons to clear the display and measurement files:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



E-5 Test Execution

With the above configurations in place, the system is prepared to perform automated file transfer and voice file quality testing.

Preparation:

1. Ensure that the Reference files specified in the VQuad Auto-Traffic Tx Provisioning and the VQT Reference File sections exist, and are of the correct (and matching) file format(s).
2. Ensure that the VQT_Degraded directory/directories specified in the VQuad Auto-Traffic Tx Provisioning section exist and are empty.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

3. Ensure that the Inventory directory/directories specified in the VQT Inventory sections exist and are empty.

E-5.1 Start VQT

1. Click the Start or Start All button to begin the test:



2. If the Start button is clicked, only the test highlighted in the list above will be started. If the Start All button is clicked, all tests in the list above will be started.

VQT is now in a 'Wait' state, and polling the Degraded directory or directories specified in the VQT Setup.

When a file is transferred into the Degraded directory by VQuad, VQT software will perform comparative testing with the degraded file and the file specified in the VQT Reference field of the VQT Setup. (Note: This should also be the file transferred by VQuad, as specified in the Tx Provisioning of VQuad Auto-Traffic Configuration.)

E-5.2 Start VQuad

In the VQuad Main Screen, click the Start All Traffic button:



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3



E-6 Transfer/Test Monitoring

E-6.1 VQuad Status Monitoring

The VQuad call and transfer progress can be monitored in two primary locations on the VQuad Main Screen:

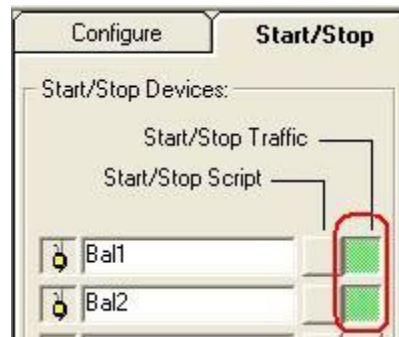
1. Device Status

The upper area of the left panel of the VQuad Main Screen will highlight the blocks associated with configured devices when they are active:



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3



2. Call Status

The lower area of the left panel of the VQuad Main Screen will indicate the status of calls in progress, and will indicate real-time transmission and receiving activity:

	Call Status	Traffic	Volume	Script
1.	Connected	Running		None
2.	Connected	Running		None

The Call Status section will also display activity of the individual devices:

Call Status Section Example 1:

	Call Status	Traffic	Volume	Script
1.	Connected	1 Rx File		None
2.	Connected	1 Rx File		None

Call Status Section Example 2:

	Call Status	Traffic	Volume	Script
1.	Connected	2 Tx File		None
2.	Connected	Running		None

E-6.2 VQT Testing Monitoring

The VQT test and transfer progress can be monitored in two primary locations in VQT:

1. The Auto Measurement screen will indicate a file count for each running test in the Count column:

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the "VQT Auto-Measurement: 10x6_Loopback" window. It features a menu bar with "File" and "Help". Below the menu is a table listing degraded directories. The table has seven columns: Degraded Directory, Reference File, Type, Option, Inventory, User ID, and Counts. There are six rows of data, each starting with a red circular icon. The "Counts" column values are 3, 3, 3, 2, 2, and 2. A red box highlights the "Counts" column header and its corresponding values.

Degraded Directory	Reference File	Type	Option	Inventory	User ID	Counts
C:\WQT_Degraded\1	C:\WQT_Reference\WQuad_Auto\Raw\fm1_...	Raw ...	Sav/All	C:\WQT_Degraded\Hold	pc1	3
C:\WQT_Degraded\2	C:\WQT_Reference\WQuad_Auto\Raw\fm1_...	Raw ...	Sav/All	C:\WQT_Degraded\Hold	pc1	3
C:\WQT_Degraded\3	C:\WQT_Reference\WQuad_Auto\Raw\fm1_...	Raw ...	Sav/All	C:\WQT_Degraded\Hold	pc1	3
C:\WQT_Degraded\4	C:\WQT_Reference\WQuad_Auto\Raw\male1...	Raw ...	Sav/All	C:\WQT_Degraded\Hold	pc1	2
C:\WQT_Degraded\5	C:\WQT_Reference\WQuad_Auto\Raw\male1...	Default	Sav/All	C:\WQT_Degraded\Hold	pc1	2
C:\WQT_Degraded\6	C:\WQT_Reference\WQuad_Auto\Raw\male1...	Default	Sav/All	C:\WQT_Degraded\Hold	pc1	2

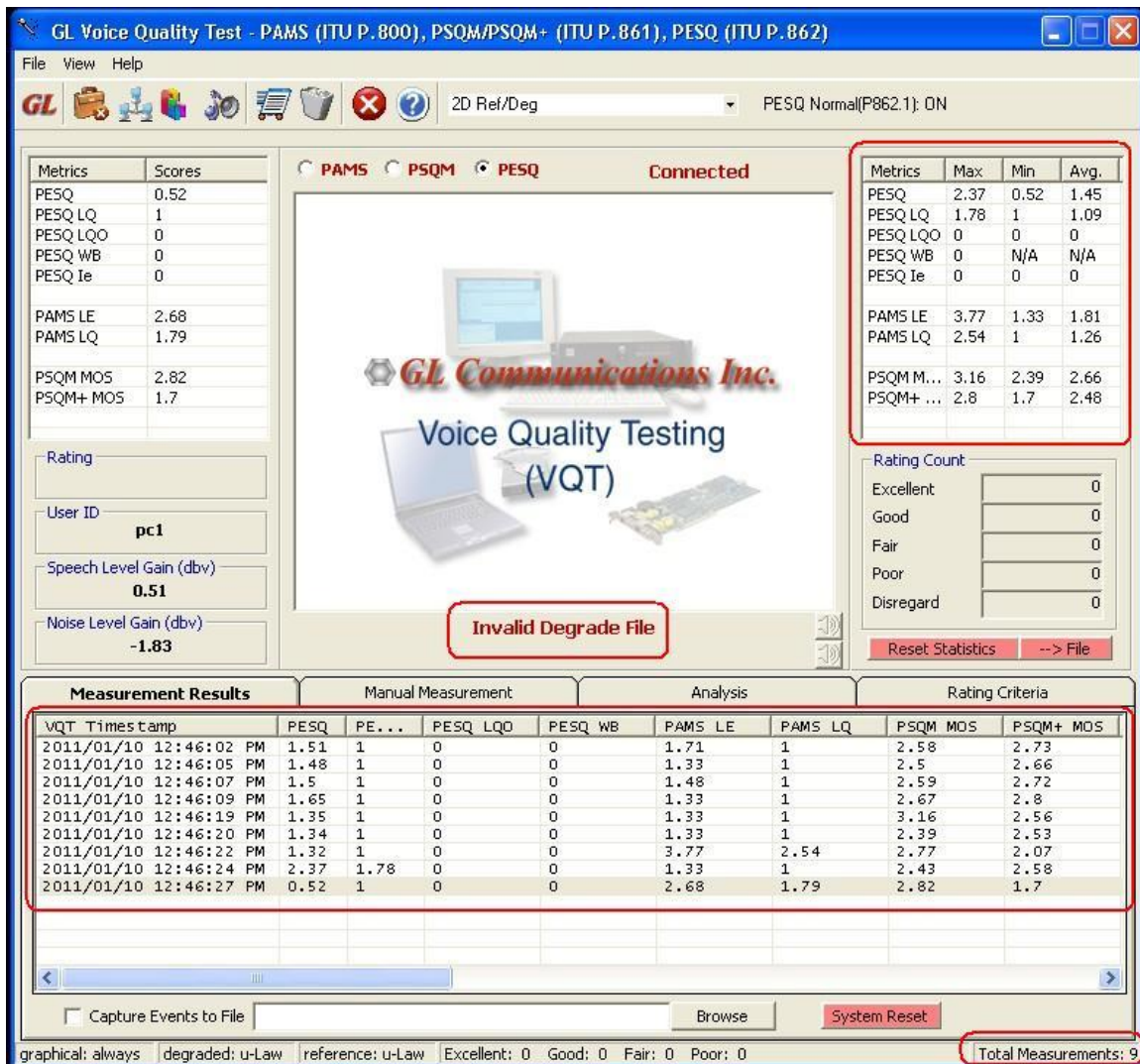
At the bottom of the window, there are two groups of buttons. The first group contains "Add", "Modify", "Delete", and "Delete All". The second group contains "Start", "Stop", "Start All", and "Stop All".

2. The VQT Main Screen will also display information about tests in progress, including any alert statuses:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



File Movement Monitoring (Windows):

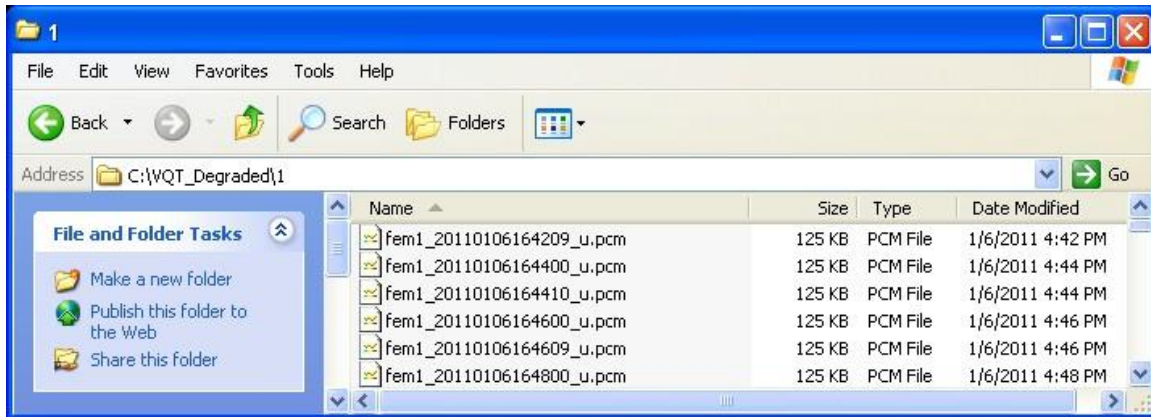
VQuad transfer progress can be monitored externally to VQuad by simply using Windows Explorer (or the Command Window) to view files as they are transferred into the VQT_Degraded directory:

VQT Degraded Directory (Explorer):

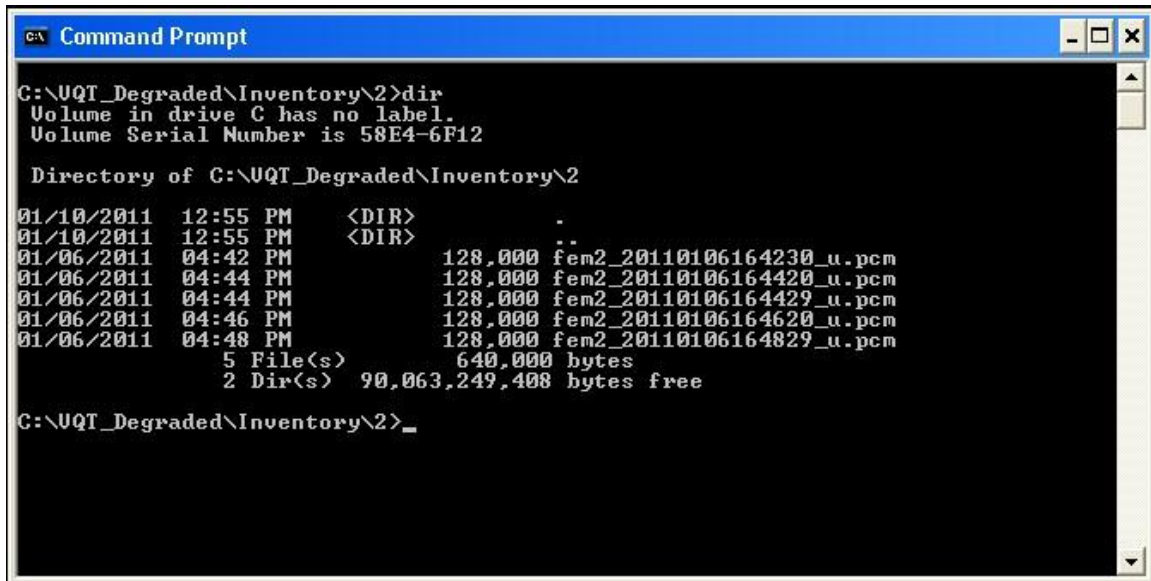


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



VQT Degraded Directory (Command Window):

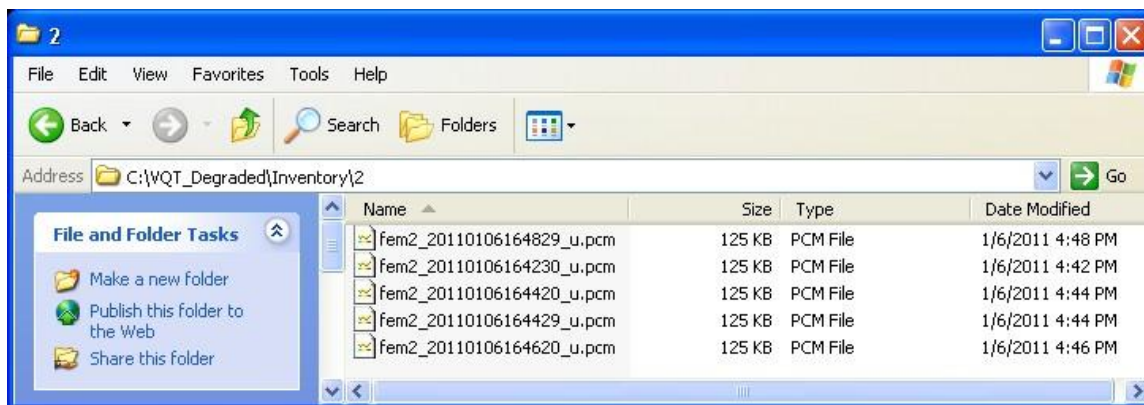


VQT progress can be monitored using Windows Explorer (or the Command Window) to view files as they are transferred from the VQT_Degraded directory into the Inventory directory:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



Note: If the VQuad transfer and VQT test operations are running simultaneously, the time files remain in the VQT_Degraded directory may be very short; depending on circumstances, they may never become visibly listed. (This will occur when VQT testing moves the files from the VQT_Degraded before Windows Explorer has updated the listing window.)

Progress can still be monitored, however, by observing the files entering the VQT Inventory directory.

E-7 Test Results

Test system hardware and software are integrated and implemented in accordance with Reference 3, and configured in accordance with the above procedures.

The results of each run are tabulated and averaged for each Date/Time Group (DTG) of the run for the following data points:

1. PAMS_LE
2. PAMS_LQ
3. PSQM
4. PSQM_MOS
5. PSQMPlus
6. Calculated PSQMPlus_MOS
7. PESQ_Score
8. PESQ_LQ
9. PESQ_MOS



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

E-7.1 Detailed Data

From Cisco *Packetcable Implementation*:

The most familiar voice quality measurement is Mean Opinion Score (MOS). MOS is expressed on a five-point scale, with 5 being the best and 1 the worst. Table MOS-1 displays further detail about MOS ratings.

Table MOS-1

Rating	Speech Quality	Distortion Level
5	Excellent	Imperceptible
4	Good	Just perceptible, not annoying
3	Fair	Perceptible, slightly annoying
2	Poor	Annoying, but not objectionable
1	Unacceptable	Very annoying, objectionable

A MOS score of 4.0 is considered "toll quality," the quality associated with traditional PSTN service. Originally, MOS scores were purely subjective, based on individuals listening to voice samples and grading them. However, because external factors (including background noise at both ends of the communication path and even the individual's mood) could greatly influence scoring, more scientific ways of quantifying voice quality have been developed.

Note: Historical research indicates that a 3 per cent packet loss results in a MOS score dropping by 0.5 (out of 5).

E-7.2 PAMS

The Perceptual Analysis/Masurement System (PAMS) was developed by British Telecom, and uses a mathematical model of human hearing. PAMS measurements report two scores: Listening Quality (LQ) and Listening Effort (LE).

These are both on a five-point scale similar to the MOS score.

Note: Research shows that PAMS scores average within a half-point when compared to traditional MOS scores.

Note: In PAMS measurements, errors in input signals are taken into consideration.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

E-7.3 PSQM and PSQM+

The Perceptual Speech Quality Measure (PSQM) was developed by the ITU (the International Telecommunications Union) as standard P.861. It is designed to measure the effect of compression on voice quality; it does not take lost packets into account.

PSQM scores are on a scale of 0 to 6.5, with 0 indicating no distortion and 6.5 indicating extreme distortion. (Thus, the lower the number, the better the voice quality.) There is no direct correlation between 'traditional' MOS scores and PSQM scores.

PSQM+ was developed to address some of these issues. PSQM+ performs additional postprocessing on PSQM values, and its results correlate more directly with 'traditional' MOS scores. It is to be noted, however, that PSQM+ still has issues with testing which involves lost packets.

The Perceptual Evaluation of Speech Quality (PESQ) is presently the most advanced method for voice quality measurement. It was designed to succeed PSQM, and is defined in ITU standard P.862.

PESQ combines the techniques of PSQM+ and PAMS, and takes into account distortion, errors, packet loss, and delay.

PESQ measurement uses a sensory model (the comparison of the original ('reference') signal with the received ('degraded') signal), and its scores are analogous to MOS scores, with the best PESQ score which can be achieved being 4.5.

E-8 Unit and Integration Verification and Validation

The specific elements tested are:

1. Two equivalent Dell Latitude D630 laptops as processing nodes (PC1 and PC2)
2. Two GL Communications Universal Telephony Agents (UTAs) (UTA155155 and UTA155156)
3. Connectors and cabling necessary for interconnection
4. GL Communications VQuad (running on both PC1 and PC2)
5. GL Communications VQT (running on either PC1 or PC2)

The tested configurations are:

1. PC1-to-UTA155155 Loopback Transfer
2. PC1-to-UTA155156 Loopback Transfer



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

3. PC2-to-UTA155155 Loopback Transfer
4. PC2-to-UTA155156 Loopback Transfer

In addition, each test will be performed with each configured User Agent (UA) in Master and in Slave mode.

The tests consist of three iterations ('runs') of each:

- 1 encoded voice file transferred 20 times
- 1 encoded voice file transferred 250 times
- 4 different encoded voice files transferred 10 times
- 4 encoded different voice files transferred 250 times
- 6 different encoded voice files transferred 10 times
- 6 different encoded voice files transferred 250 times

All transfers are bidirectional, with each DUT transmitting and receiving.



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

References

1. Department of Defense Unified Capabilities Requirements 2008 (UCR 2008)
2. Fusion *Voice Engine User's Manual*, Software Version 3.2.X, Part Number/Version: FVE V3.2.X, Release Date: November, 2010
3. *GL Communications Voice Quality Testing (VQT) User Guide*, Updated June 2010
4. *GL Communications VQuad™ (VOICE Quad) User Manual*, Updated May 2008
5. *GL Communications GL VQT Reference*, Log Output, September 2004
6. *Packetcable Implementation*, Jeff Riddel, Copyright 2007



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

APPENDIX F Real-Time Protocol (RTP) Testing



**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

Maritime Systems

Table of Contents

F-1	OVERVIEW AND ASSUMPTIONS	309
F-1.1	Overview	309
F-1.2	Voice File Transfer and Analysis and Assumptions	309
F-2	RTP TRANSFER TESTING.....	310
F-2.1	Detailed Implementation	310
F-2.2	Hardware Setup.....	310
F-2.3	Data Path.....	311
F-2.4	Software Setup	311
F-2.4.1	VQuad Setup	311
F-2.4.2	Modify or Create a VQuad Device Configuration for RTP Testing	314
F-2.4.3	Auto-Traffic Configuration	316
F-2.4.4	Configure Device 1	317
F-2.5	Modify or Create a VQuad Auto Traffic Configuration for RTP Testing	321
F-2.6	VQT Setup.....	323
F-2.7	Checking VQT Auto-Measurement Setup	329
F-3	TEST EXECUTION	331
F-3.1	Preparation.....	331
F-3.2	Start VQT.....	332
F-3.3	Start VQuad	332
F-4	TRANSFER/TEST MONITORING	333
F-4.1	VQuad Status Monitoring	333
F-4.2	VQT Testing Monitoring.....	334
F-5	TEST RESULTS	338
F-5.1	PAMS.....	339
F-5.2	PSQM and PSQM+	340
F-5.3	PESQ	340
F-6	UNIT AND INTEGRATION VERIFICATION AND VALIDATION	340



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

F-1 Overview and Assumptions

F-1.1 Overview

The purpose of this document is to detail the setup, implementation, and execution of testing voice quality in the use of the L-3 Maritime-developed SIP solution in progress.

The voice quality testing for the project will be done in three parts:

1. Baseline Verification ensures the functionality of individual components and test elements, and includes verifying:

- Equipment/setup
- Software/setup
- Test files

2. RTP/Transfer Testing evaluates the performance of the Real-Time Protocol (RTP) to be implemented in the integrated solution. It consists of:

- Establishing communications links over RTP
- Transferring selected voice files over the link
- Evaluating the degradation of files transferred

3. End-to-End Communication Verification tests the VoIP/AS-SIP stack (for call control) and the RTP (for data/voice transfer), and the selected endpoint nodes as an integrated system, and consists of:

- Call completion and traffic handling analysis and evaluation

F-1.2 Voice File Transfer and Analysis and Evaluation Assumptions

- Voice files will be transferred and analyzed using u-Law encoding as the standard.
- GL Communications VQT software will be used to evaluate the integrity and parameters of transferred voice files.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

F-2 RTP Transfer Testing

F-2.1 Detailed Implementation

The baseline for the testing system is defined as the hardware, software, configuration, and files used for testing and verification of the VoIP/SIP solution. Following are the details of the present and planned elements of the baseline.

This implementation tests the Real-Time Protocol (RTP) to be implemented in the integrated solution. It incorporates the files and VQuad functionality of earlier tests, and introduces the element of device-to-device voice file transfer across a network using the developed protocol.

- VoIP/AS-SIP stack (server)
- RTP (endpoint-to-endpoint)
- Endpoint devices

F-2.2 Hardware Setup

The testing system hardware consists of:

- Two equivalent Dell Latitude D630 laptops as processing nodes
- Connectors and cabling necessary for interconnection

Figure 1 illustrates the equipment setup for the above transfers.

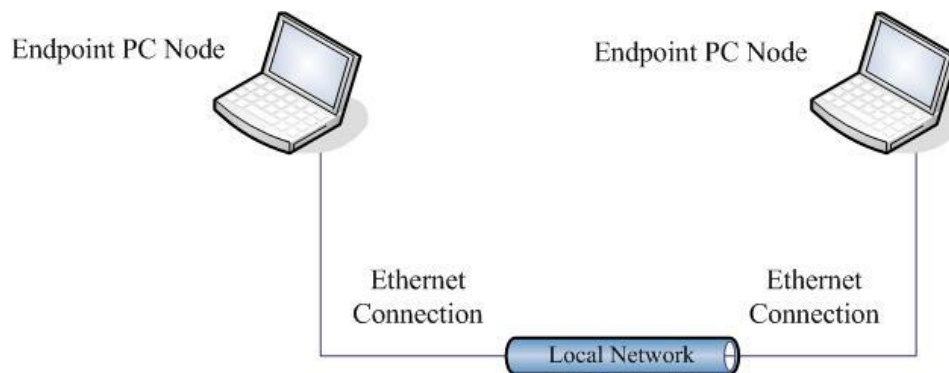


Figure 1 PC-to-PC Transfer Test Setup



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

In addition, each test will be performed with each configured User Agent (UA) in Master and in Slave mode.

Two-Endpoint-Node RTP Test Setup:

The configuration used for the RTP tests is a two-PC Endpoint Node system.

To configure hardware for RTP testing:

1. Connect the VoIP Phone Set 1 and VoIP Phone Set 2 network connections to the local Ethernet network via Category 5 cables.

F-2.3 Data Path

Selected voice files will be transferred from Endpoint PC Node 1 to Endpoint PC Node 2 and stored in the local VQT_Degraded directories (see paragraph F-2.4.1, Software Setup\VQuad Configuration).

F-2.4 Software Setup

The testing and analysis system software consists of:

Windows XP SP3 (operating system for processing nodes)

GL Communications VQuad (file transfer and traffic generation software)

GL Communications VQT (Voice Quality Testing and analysis software)

Wireshark and other utilities necessary for support and analysis

Microsoft Excel and Word for data analysis and presentation

F-2.4.1 VQuad Setup

To configure GL Communications VQuad for RTP/Transfer Testing:

1. Start VQT by double-clicking the GL VQuad icon:

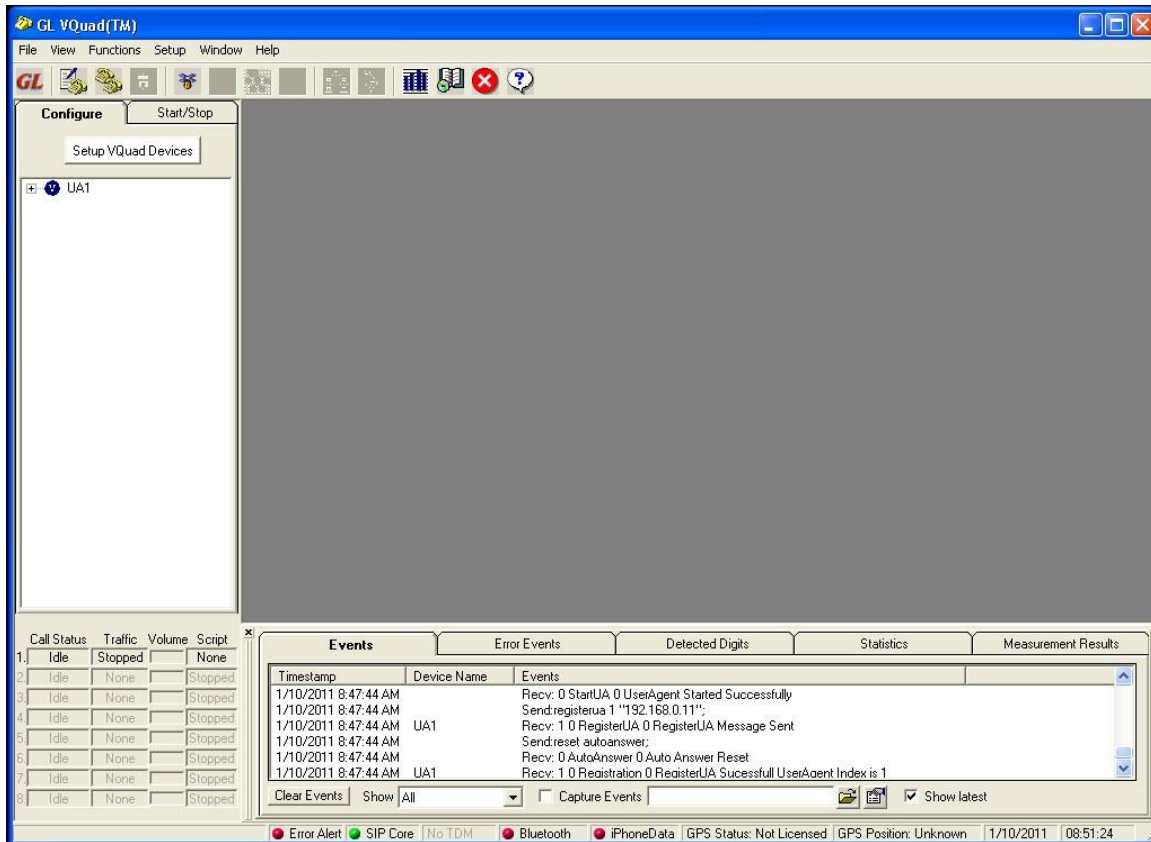


2. The VQuad Main Screen appears.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



VQuad Device Configuration

- Click the Configure tab:



A custom device configuration has been created for this test implementation. To access and use the configuration, click the Setup VQuad Devices button:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



4. Click the Devices Configuration – Load button:



From the list of available configurations, select PCx_Transfer_20x1.

The naming scheme for device configurations is as follows:

For the configuration PCx_Transfer_20x1:

- PCx_ indicates the PC testing node for which the configuration is designed:
- PC1 indicates Endpoint PC Node 1
- PC2 indicates Endpoint PC Node 2
- PCx indicates the configuration can be used for either Endpoint PC Node

Transfer_ indicates the type of test:

- Loopback indicates the test is single-node, endpoint-to-same-endpoint
- Transfer indicates the test is multi-node (two or more nodes), transfer endpoints different
- EndToEnd indicates that all devices in a VoIP/AS-SIP communication path will be tested

20x1 indicates the number of transfers and the number of files:

- The first number (in this case, 20) is the total number of transfers to be performed



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- The second number (in this case, 1) is the number of files which will be transferred

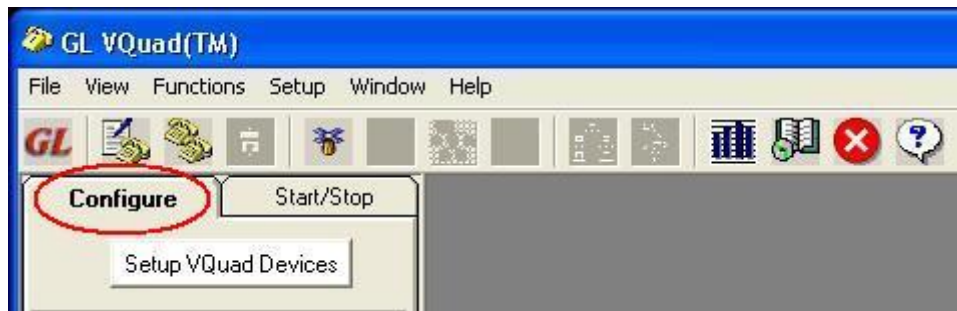
Thus, the configuration name PC2_Transfer_100x4 would indicate:

- The configuration is designed for PC2
- It is a multi-node transfer configuration
- 100 transfers of four files will be performed

F-2.4.2 Modify or Create a VQuad Device Configuration for RTP Testing

If the predefined configuration is not available, or another configuration is to be created, use the following procedure:

1. Click the Device Configure tab:



2. Click the Setup VQuad Devices button:



For RTP testing, two separate devices will be set up as 'VoIP'.

3. In the GL VQuad™ Device Configuration window, click the Number of Devices down arrow and select 1.

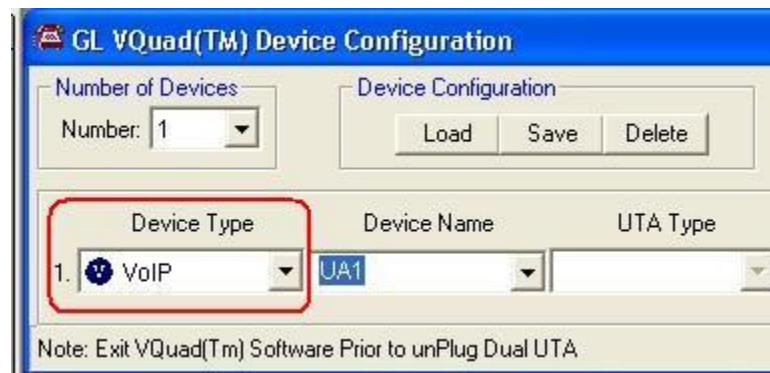


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

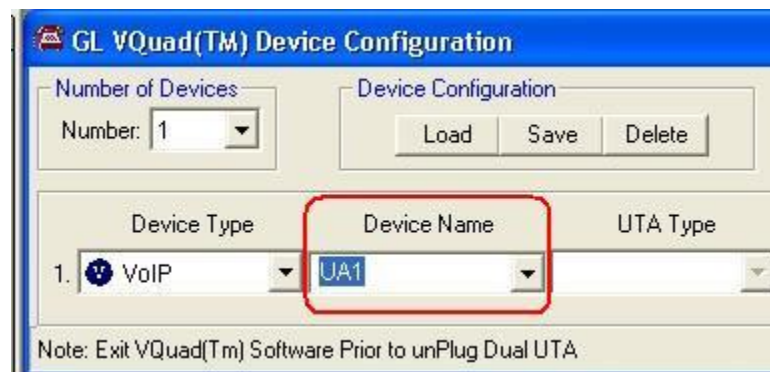
Maritime Systems



- Click the Device Type down arrow and select VoIP as the device type:



- Enter a name in the Device Name field. For illustration purposes, the Device Name entered will be UA1:

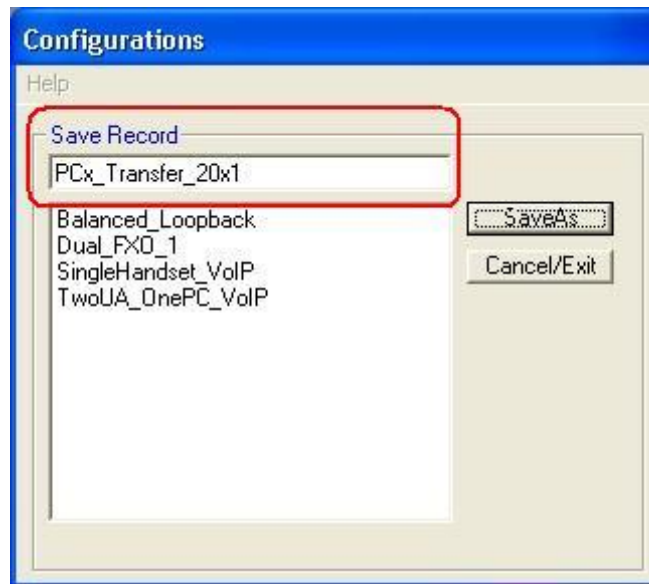


- Save the device configuration with a descriptive name using the Save button. For illustration purposes, the configuration will be saved as PCx_Transfer_1:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



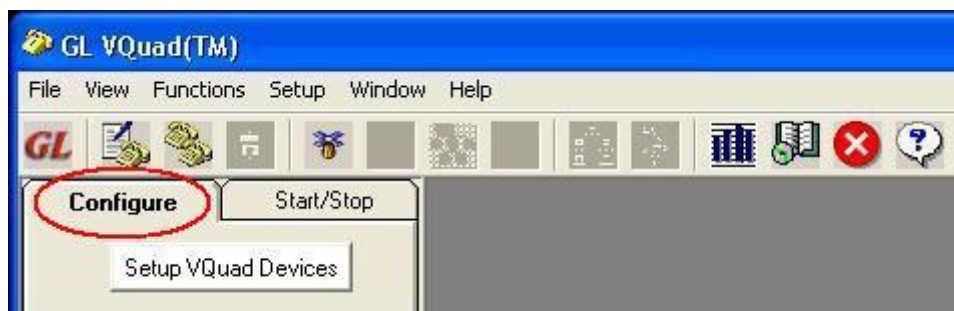
VQuad devices are now configured to transfer files using the configured VoIP end points via Ethernet.

F-2.4.3 Auto-Traffic Configuration

Auto-Traffic Configuration determines the method of transfer (master/slave identification), the type of file transferred, and the locations of files transmitted and received.

Custom Auto-Traffic Configurations have been created for this test implementation. To access and use the configurations, use the following procedure:

In the VQuad Main Window, click the Configure tab.





Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

F-2.4.4 Configure Device 1

1. Double-click UA1 (or any other name configured to Device 1 in the previous procedure) in the Devices panel on the left of the VQuad Main Window. Alternately, click the + sign to the left of the named device:



The UA1 detail categories will be listed:



Double-click the Auto-Traffic entry:



The Auto-Traffic Configuration window will appear:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Auto Traffic Configuration - Auto Created

Device Name: Bal1

☐ Auto Created Traffic Configuration ☒ Use Pre-defined Custom Configuration

Use Pre-defined Custom Configuration

Select Custom Config: 1000x4x2_timed_raw_master1 Show Detail

Configuration: Auto Created Association Device: Bal1

2. Select UA1 as the Device Name:

Auto Traffic Configuration - Auto Created

Device Name: UA1

☐ Auto Created Traffic Configuration ☒ Use Pre-defined Custom Configuration

Use Pre-defined Custom Configuration

Select Custom Config: PCx_Transfer_20x1_Master Show Detail

Configuration: Auto Created Association Device: UA1

3. Click the Use Pre-defined Custom Configuration radio button:

Auto Traffic Configuration - Auto Created

Device Name: UA1

☐ Auto Created Traffic Configuration ☒ Use Pre-defined Custom Configuration

Use Pre-defined Custom Configuration

Select Custom Config: PCx_Transfer_20x1_Master Show Detail

Configuration: Auto Created Association Device: UA1

4. Click the down arrow to the right of the Configuration Name field to access the list of available configurations.
5. Select the configuration PCx_Transfer_20x1 for Device 1.
6. Ensure that the following parameters are correct for the relevant device:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'Auto Traffic Configuration - PCx_Transfer_20x1' window with the 'Auto Trigger' tab selected. The 'Master' sub-tab is active. The configuration includes:

- Trigger On Time** (selected):
 - Cycle Time (s): 30
 - Loop Period (s): 120
 - Stop Iterations: 20
 - File Length (s): 8
 - Tx/Rx Offset (s): 6
 - Send Delay (ms): 1500
- Trigger On DTMF digits** (unselected)
- Trigger On MF digits** (unselected)
- Trigger On User Defined Tones** (unselected)

Buttons at the bottom: Load Configuration, Save As Configuration. Status bar: Configuration: PCx_Transfer_20x1, Association Device: UA1.

PCx_Transfer_20x1 Auto Traffic Tab

The screenshot shows the 'Auto Traffic Configuration - PCx_Transfer_20x1' window with the 'General' tab selected. The configuration includes:

- File Format**:
 - 8000; 8-bit; A-Law (unselected)
 - 8000; 8-bit; Mu-Law (unselected)
 - 8000; 16-bit; PCM (selected)
- Increment Rx File Name**:
 - Sequential (unselected)
 - Time Stamp (selected)
 - GPS + Time Stamp / ITS + Time Stamp (unselected)
- Digit Parameters**:
 - Enable Digit Tx: ☐ On Time (ms): 200 High Power (-dB): 10
 - Enable Multiple Digit Detection: ☐ Off Time (ms): 750 Low Power (-dB): 10

Buttons at the bottom: Load Configuration, Save As Configuration. Status bar: Configuration: PCx_Transfer_20x1, Association Device: UA1.

PCx_Transfer_20x1 General Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The screenshot shows the 'Tx Provisioning' tab of the 'Auto Traffic Configuration - PCx_Transfer_20x1' application. The interface includes a 'Timing' column with values 0, 30, 60, and 90. A 'Reference File Path' column contains the path 'C:\WQT_Reference\WQuad_Auto\Raw\mem1_1.pcm'. To the right, there are input fields for 'Cycle Time(ms)' (2000), 'Identifier Digit' (with 'In File' and 'Generate' radio buttons), 'Power (dB)' (0), and 'Duration (ms)' (0). At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons, and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

PCx_Transfer_20x1 Tx Provisioning Tab

The screenshot shows the 'Rx Provisioning' tab of the 'Auto Traffic Configuration - PCx_Transfer_20x1' application. The interface includes a 'Timing' column with values 8, 38, 68, and 98. A 'Degraded File Path' column contains the path 'C:\WQT_Degraded\1\mem1'. To the right, there are input fields for 'Time (ms)' with values 7000, 7000, 7000, 7000, 7000, 7000, 0, 0, 0, and 0. At the bottom, there are 'Load Configuration' and 'Save As Configuration' buttons, and a status bar showing 'Configuration: PCx_Transfer_20x1' and 'Association Device: UA1'.

PCx_Transfer_20x1 Rx Provisioning Tab



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

F-2.5 Modify or Create a VQuad Auto Traffic Configuration for RTP Testing

If the existing configuration is not available or is not correct for the required test, use the following procedure to modify it or create a new one:

Configure the Auto Trigger, General, Tx Provisioning, and Rx Provisioning tabs in accordance with applicable requirements.

Guidelines for configuring the fields of the tabs follow:

Auto Trigger Tab Fields:

- **Cycle Time:** This is the time allotted for the VQuad™ master and slave operations to take place. This time should be adequately long when bidirectional operations are performed.
- **Loop Period:** This is the total length of a timed cycle. (The default profile uses six files in a cycle.)
- **Stop Iteration:** Specifies the number of iterations (cycles multiplied by number of lines configured) that the VQuad™ will execute before stopping the test.
- **File Length:** This indicates the recording duration and is required to set up the VQuad™ time triggering flow. This includes the time to receive an entire file length plus the send delay time.
- **Tx/Rx Offset:** This is the ‘wait period’ after completion of a Tx action, but before the return of Tx action for bi-directional file transfer.
- **Send Delay:** This is the time after which the VQuad™ receiving end starts recording. This is a buffer time which helps counter the clock variations of two independent PCs.

F-2.5.1 General Tab Fields

- **File Format:** This setting selects the format of the file that the VQuad™ receiver will be saved to after the receiver completes recording of an incoming degraded file. The recorded degraded file type must match the type of VQuad™ reference file being sent, or the VQT Analysis program will yield very bad VQT scores due to mismatched file types. VQuad™ reference files are provided in linear PCM, A-law, and Mu-law encoding formats. There are usually small differences in VQT scores when using one file type as opposed to another. VQuad™ normally executes using linear PCM Reference files.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- Note: When performing transfers using Dual UTAs, the file type must be 16-bit 8000 Hz sampled, Little Endian (Intel) PCM.
- Increment Rx File Name : When the VQuad™ completes recording of an incoming degraded voice file, the file several options can be used to suffix the received file name to simplify identifying the degraded files.

The filename prefix for each recorded sample is defined by the filename rules setup in the Rx Provisioning tab.

- Sequential: The sequential option simply appends a sequential number to each sample filename in the order as they are received. By default, the numbering starts from zero, but the user can select Sequential File Numbering to start the numbering sequence at any number, or to reset a sequence that was previously started.
- Time Stamp: Each incoming/recorded-degraded file has the time stamp (from the PC clock) that the sample was received as the filename suffix. (Note: Ensure that the PC clock is set to the correct time.)
- Digit Parameters
 - The Digit/Tone Parameters only work in conjunction with the User-defined Tone Triggering method when trigger tones external to the sample voice file are required.
 - If Enable Digit TX is not checked, the VQuad™ program uses MF, DTMF or User-defined tones, attached (prefixed) to the VQT Reference File, to trigger the recording and to identify the VQT Reference File that follows. If Enable Digit TX is checked, the VQuad™ program sends tones according to the table of user-defined tones before sending voice file. In this case, the reference voice files without prefixed tones must be used. (Refer to section Tone/Digit Method for Sending/Recording for details)
 - Enable Multiple Digit Detection is used only in 'master mode' with DTMF/MF trigger. It allows multiple digits detection during one session.
 - For example, in normal case (Enable Multiple Digit Detection is not checked), one session consists of a 'TX file' and a 'RX file'. Even if it detects another digit, it will wait till the end of the cycle duration and then proceeds to the



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

next session. However, the Enable Multiple Digit Detection option when checked, will again allow “RX” file event to occur after another digit is detected.

- Enter On Time (200ms or longer), Off Time (750ms or longer), High Power (-dB) and Low Power (-dB) values.

F-2.5.2 Tx Provisioning Tab Fields

- Reference File Path: This field determines the location of the file(s) to be transferred.

Sample files are provided by GL Communications, and are located in local subdirectories of C:\VQT_Reference\.

F-2.5.3 Rx Provisioning Tab Fields

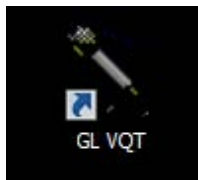
- Degraded File Path: The path on the local PC which is the destination for files transferred in the assigned time slot.

The final characters of the Degraded File Path entry will become the initial characters of the files placed in the Degraded File Path directory. These characters will be suffixed with the Sequential or Time Stamp data (configured in the General tab).

F-2.6 VQT Setup

To configure GL Communications VQT for Auto Measurement Setup and Execution:

1. Start VQT by double-clicking the GL Voice Quality Testing icon:



2. The VQT Main Screen appears.

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

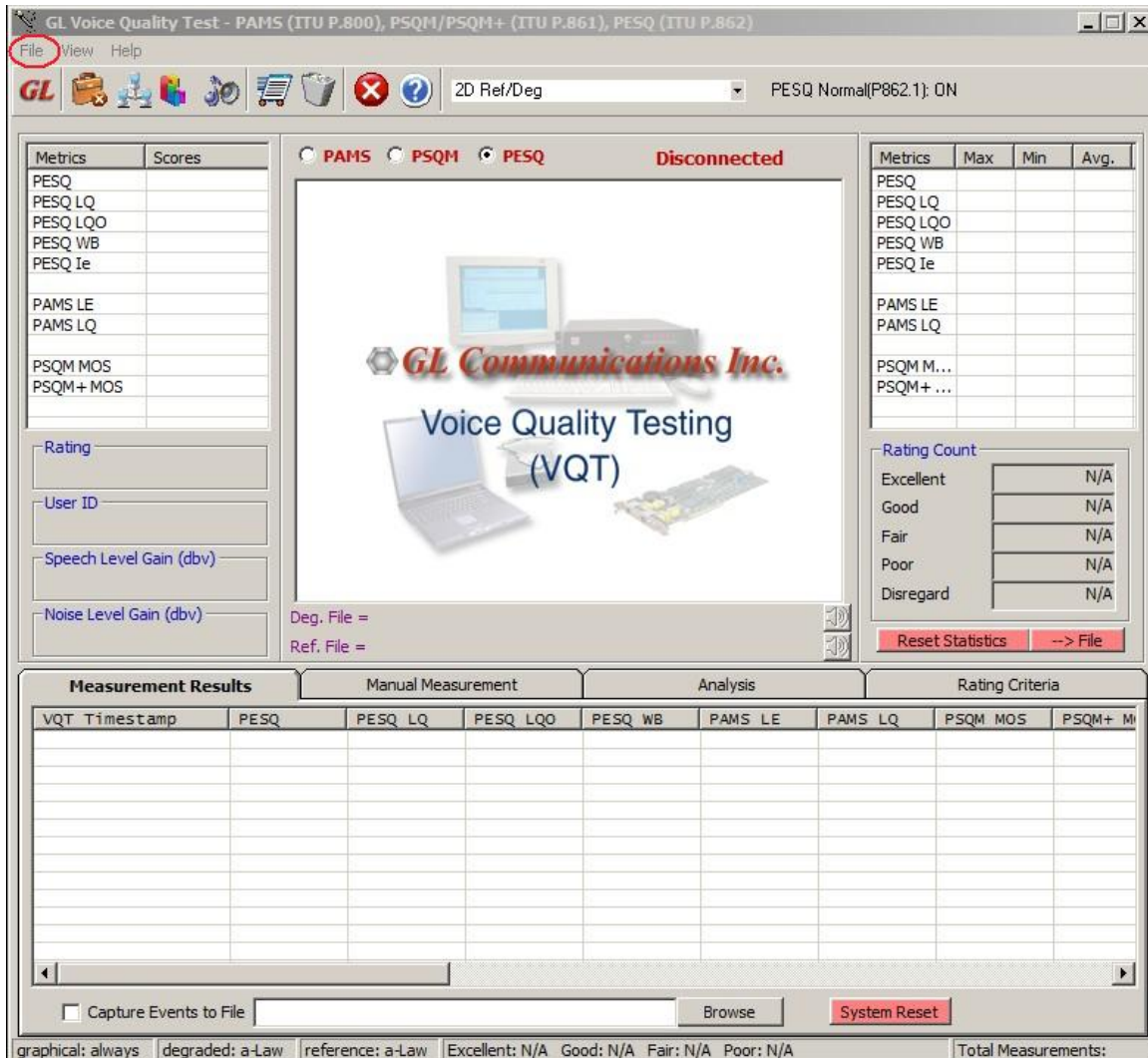
[illegible]

3. From the VQT main screen, select File > Auto Measurement:

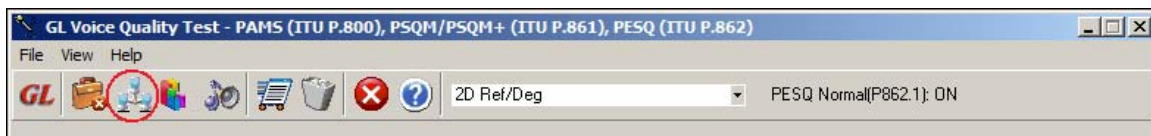


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



4. Alternatively, the Auto-Measurement icon can be clicked:



5. The VQT Auto-Measurement screen appears:

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

[illegible]

6. Click the Add button:

The screenshot shows the "QVT Auto-Measurement" application window. At the top is a menu bar with "File" and "Help". Below the menu bar is a large table with seven columns: "Degraded Directory", "Reference File", "Type", "Option", "Inventory", "User ID", and "Counts". The table has approximately 10 rows, all of which are currently empty. At the bottom of the window is a control panel containing several buttons. On the left side of this panel, there are four buttons: "Add", "Modify", "Delete", and "Delete All". The "Add" button is highlighted with a red circle. To the right of these buttons are four more buttons: "Start", "Stop", "Start All", and "Stop All".

7. Populate the indicated fields to set up VQT Auto-Measurement:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

8. Enter the Degraded Directory in the space provided. This is the directory containing the files transferred in the VQuad RTP Test, and is configured in VQuad Auto Traffic Configuration. (See VQuad Auto Traffic Configuration – Rx Provisioning Tab.)

For illustration purposes, C:\VQT_Degraded\1 will be entered.

Note: Ensure that the directory C:\VQT_Degraded\1 (or whatever directory is specified as the Degraded Directory) exists on the hard drive. Clicking the Open Folder icon will open a Windows selection dialogue which will allow selection from existing directories:

9. Enter the Reference File in the space provided. This is the original file which was transferred according the VQuad RTP Test, and is configured in VQuad Auto Traffic Configuration. (See VQuad Auto Traffic Configuration – Tx Provisioning Tab.)



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

For illustration purposes, C:\VQT_Reference\VQuad_Auto\fem1.pcm will be selected.

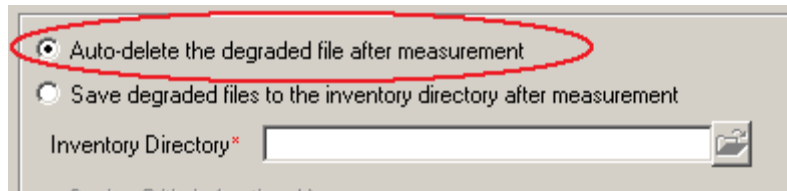
10. Select the Save degraded files to an inventory directory after measurement radio button.

11. Enter the Inventory Directory in the space provided. This is the directory to which files will be transferred after testing. Thus, the file path is:

For illustration purposes C:\VQT_Degraded\Inventory\1\ will be selected.

Note: Ensure that C:\VQT_Degraded\1 exists on the hard drive. Clicking the Open Folder icon will open a Windows selection dialogue which will allow selection from existing directories.

12. Alternately, the Auto-delete the degraded file after measurement radio button can be selected:



If this button is selected, the transferred files will be deleted from the Degraded directory after testing. This option may be preferred if the test parameters are known correct, and the test measurements will not be repeated.

However, for purposes of data retention, or if running any given tests again is necessary or possible, the use of the Save degraded files to an inventory directory after measurement is preferable.

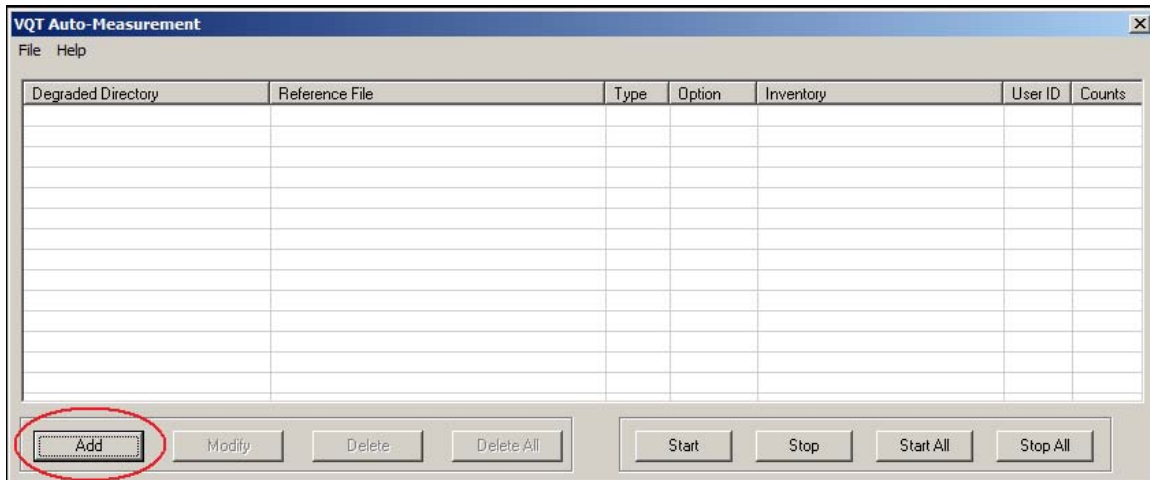
Within the User ID field, enter a User ID for tracking purposes. This field is not verified or cross-checked; it is stored as part of the data produced by the test, and can be useful for granular data analysis in a multi-testing environment.

13. Select the Add button to add the session:

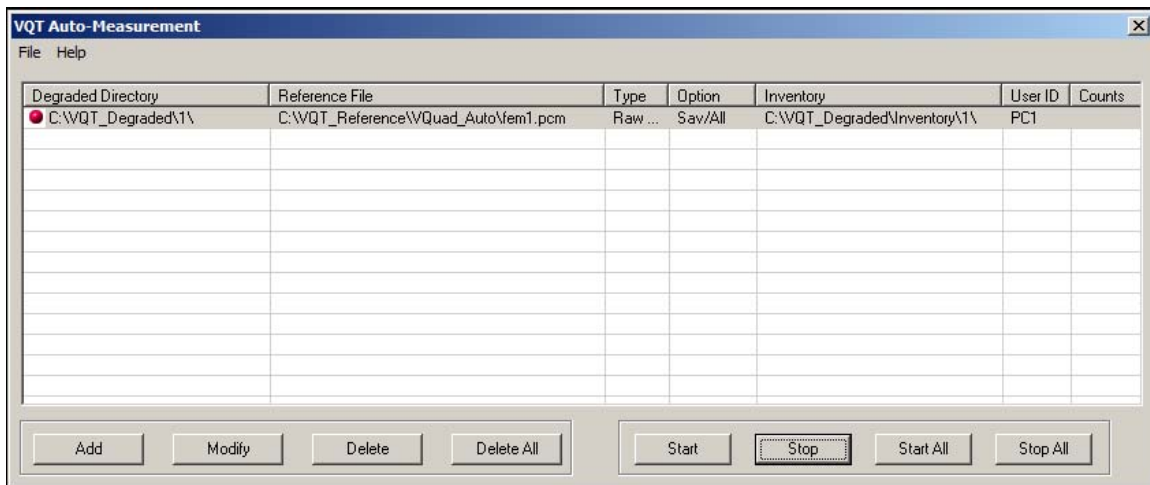


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



14. The VQT Auto-Measurement window will now indicate the data associated with the test entered:



F-2.7 Checking VQT Auto-Measurement Setup

To verify that the VQT setup is functional, manually copy a file of the correct format into the specified Degraded directory. If setup is correct, VQT will automatically run the measurement on the file and move the file to the Inventory directory.

1/ Click the Start or Start All button to begin the test:



2/ If the Start button is clicked, only the test highlighted in the list above will be started. If the Start All button is clicked, all tests in the list above will be started.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

If the specified Degraded directory contains files when VQT testing is started, VQT will immediately begin processing the files. The quality tests will be performed and tabulated, then the files tested will be moved to the specified Inventory directory.

If the specified Degraded directory does not contain files when VQT testing was started, will wait for a file to be placed in the directory. When a file is placed in the directory (usually by VQuad transfer), the VQT will automatically perform the measurement and move the file to the specified Inventory directory.

3. When the tests have been verified to run correctly, click the Stop All button to end testing:

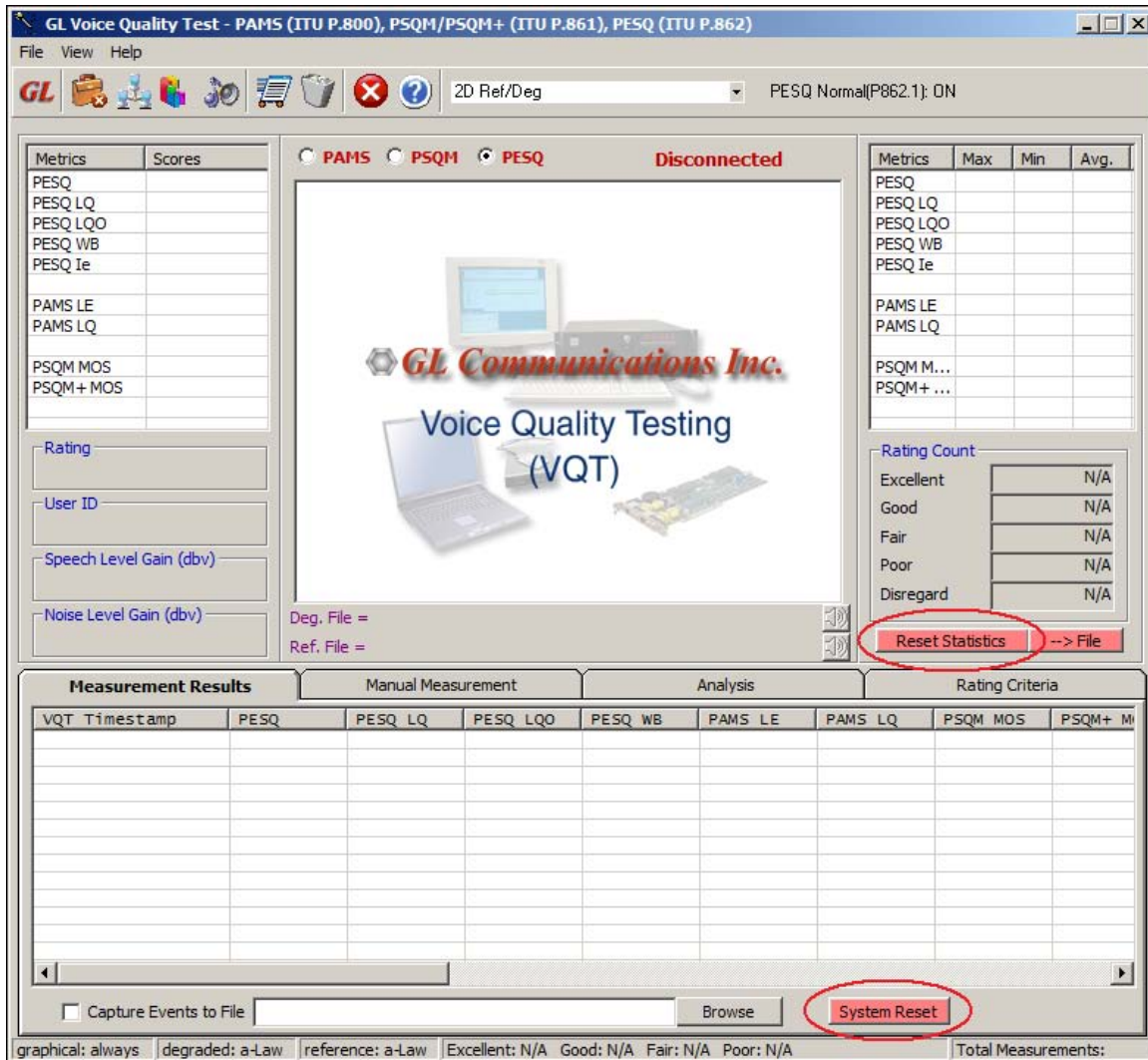


Before starting a test on transferred data, click the Reset Statistics and Reset System buttons to clear the display and measurement files:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



F-3 Test Execution

With the above configurations in place, the system is prepared to perform automated file transfer and voice file quality testing.

F-3.1 Preparation

1. Ensure that the Reference files specified in the VQuad Auto-Traffic Tx Provisioning and the VQT Reference File sections exist, and are of the correct (and matching) file format(s).
2. Ensure that the VQT_Degraded directory/directories specified in the VQuad Auto-Traffic Tx Provisioning section exist and are empty.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

3. Ensure that the Inventory directory/directories specified in the VQT Inventory sections exist and are empty.

F-3.2 Start VQT

1. Click the Start or Start All button to begin the test:



2. If the Start button is clicked, only the test highlighted in the list above will be started. If the Start All button is clicked, all tests in the list above will be started.

VQT is now in a 'Wait' state, and polling the Degraded directory or directories specified in the VQT Setup.

When a file is transferred into the Degraded directory by VQuad, VQT software will perform comparative testing with the degraded file and the file specified in the VQT Reference field of the VQT Setup. (Note: This should also be the file transferred by VQuad, as specified in the Tx Provisioning of VQuad Auto-Traffic Configuration.

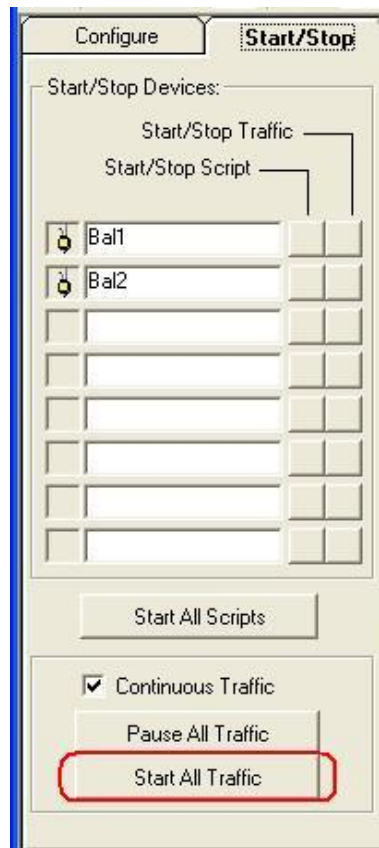
F-3.3 Start VQuad

1. In the VQuad Main Screen, click the Start All Traffic button:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



2. When file transfers are completed, click Stop All Traffic button from the VQuad™ Control Panel:



F-4 Transfer/Test Monitoring

F-4.1 VQuad Status Monitoring

The VQuad call and transfer progress can be monitored in two primary locations on the VQuad Main Screen:

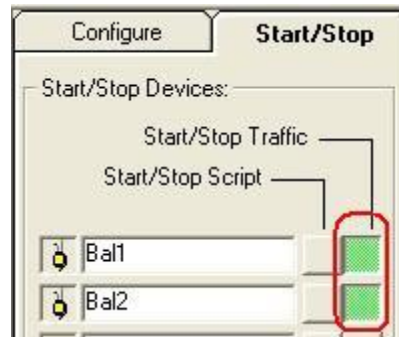
1. Device Status



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The upper area of the left panel of the VQuad Main Screen will highlight the blocks associated with configured devices when they are active:



2. Call Status

The lower area of the left panel of the VQuad Main Screen will indicate the status of calls in progress, and will indicate real-time transmission and receiving activity:

	Call Status	Traffic	Volume	Script
1.	Connected	Running		None
2.	Connected	Running		None

The Call Status section will also display activity of the individual devices:

Call Status Section Example 1

	Call Status	Traffic	Volume	Script
1.	Connected	1 Rx File		None
2.	Connected	1 Rx File		None

Call Status Section Example 2

	Call Status	Traffic	Volume	Script
1.	Connected	2 Tx File		None
2.	Connected	Running		None

F-4.2 VQT Testing Monitoring

The VQT test and transfer progress can be monitored in two primary locations in VQT:

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

1. The Auto Measurement screen will indicate a file count for each running test in the Count column:

The screenshot shows the 'VQT Auto-Measurement: 10x6_Loopback' window. It contains a table with the following data:

Degraded Directory	Reference File	Type	Option	Inventory	User ID	Counts
● C:\WQT_Degraded\1	C:\WQT_Reference\WQuad_Auto\Raw\fm1_...	Raw ...	Sav/All	C:\WQT_Degraded\Hold	pc1	3
● C:\WQT_Degraded\2	C:\WQT_Reference\WQuad_Auto\Raw\fm1_...	Raw ...	Sav/All	C:\WQT_Degraded\Hold	pc1	3
● C:\WQT_Degraded\3	C:\WQT_Reference\WQuad_Auto\Raw\fm1_...	Raw ...	Sav/All	C:\WQT_Degraded\Hold	pc1	3
● C:\WQT_Degraded\4	C:\WQT_Reference\WQuad_Auto\Raw\male1...	Raw ...	Sav/All	C:\WQT_Degraded\Hold	pc1	2
● C:\WQT_Degraded\5	C:\WQT_Reference\WQuad_Auto\Raw\male1...	Default	Sav/All	C:\WQT_Degraded\Hold	pc1	2
● C:\WQT_Degraded\6	C:\WQT_Reference\WQuad_Auto\Raw\male1...	Default	Sav/All	C:\WQT_Degraded\Hold	pc1	2

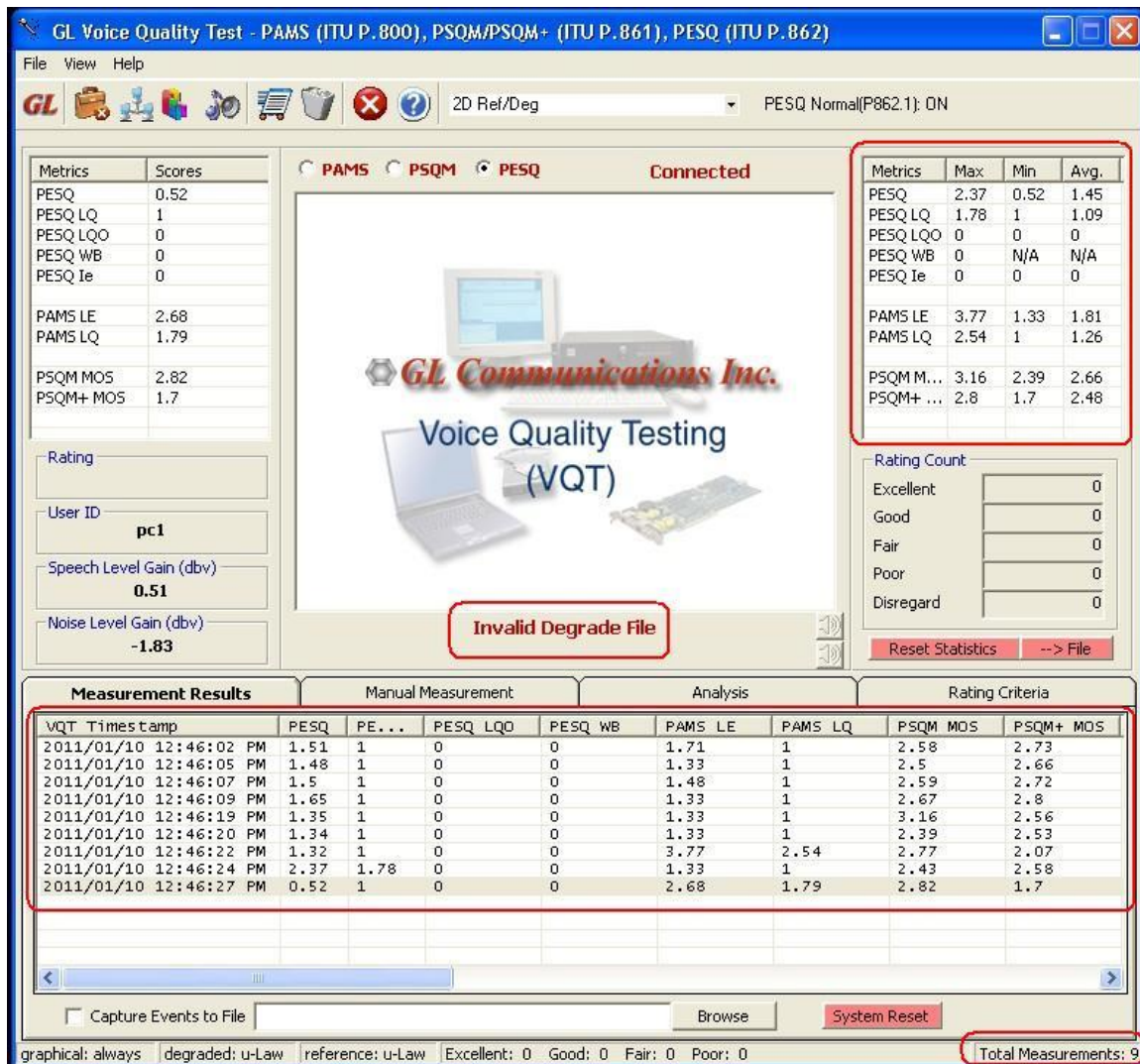
At the bottom of the window, there are buttons for 'Add', 'Modify', 'Delete', 'Delete All', 'Start', 'Stop', 'Start All', and 'Stop All'.

2. The VQT Main Screen will also display information about tests in progress, including any alert statuses:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



File Movement Monitoring (Windows):

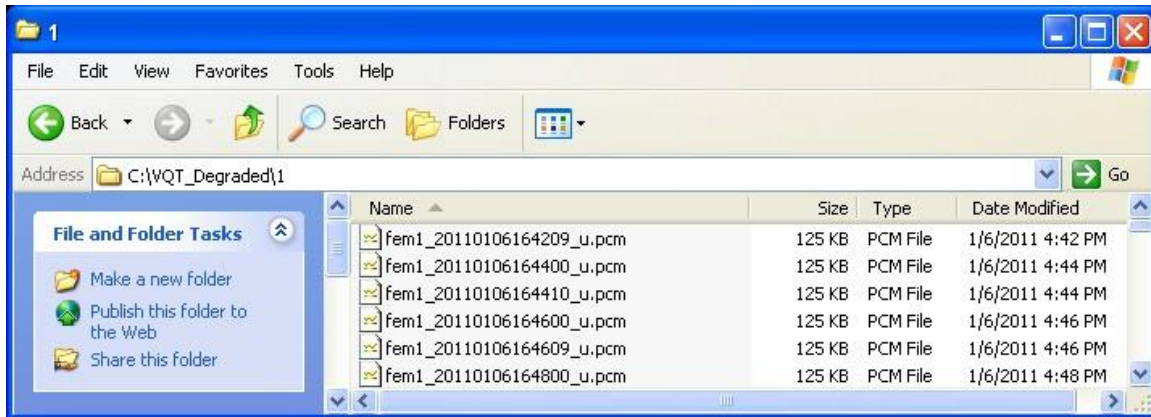
VQuad transfer progress can be monitored externally to VQuad by simply using Windows Explorer (or the Command Window) to view files as they are transferred into the VQT_Degraded directory:

VQT Degraded Directory (Explorer):

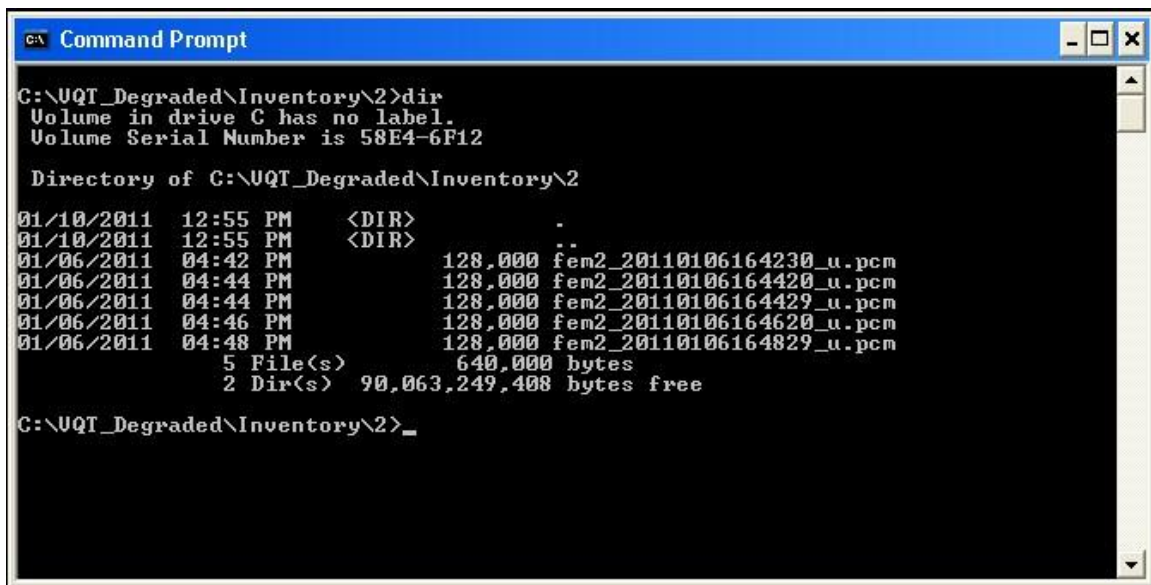


Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



VQT Degraded Directory (Command Line):

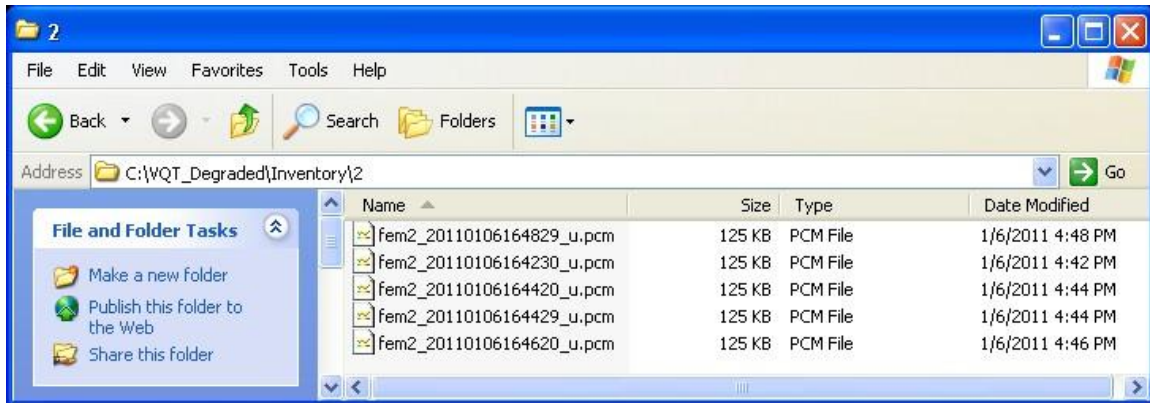


VQT progress can be monitored by Windows Explorer (or the Command Window) to view files as they are transferred from the VQT_Degraded directory into the Inventory directory:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems



Note: If the transfer and test operations are running simultaneously, the time files remain in the VQT_Degraded directory may be very short; depending on circumstances, they may never become visibly listed. (This will occur when VQT testing moves the files from the VQT_Degraded before Windows Explorer has updated the listing.)

Progress can still be monitored, however, by observing the files entering the VQT Inventory directory.

F-5 Test Results

Test system hardware and software are integrated and implemented in accordance with Reference 3, and configured in accordance with the above procedures.

The summary of testing for each of the configurations and transfers is shown in Figure P1.

From *Cisco Packetcable Implementation*:

The most familiar voice quality measurement is Mean Opinion Score (MOS). MOS is expressed on a five-point scale, with 5 being the best and 1 the worst. Table MOS-1 displays further detail about MOS ratings.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Table MOS-1

Rating	Speech Quality	Distortion Level
5	Excellent	Imperceptible
4	Good	Just perceptible, not annoying
3	Fair	Perceptible, slightly annoying
2	Poor	Annoying, but not objectionable
1	Unacceptable	Very annoying, objectionable

A MOS score of 4.0 is considered "toll quality," the quality associated with traditional PSTN service. Originally, MOS scores were purely subjective, based on individuals listening to voice samples and grading them. However, because external factors (including background noise at both ends of the communication path and even the individual's mood) could greatly influence scoring, more scientific ways of quantifying voice quality have been developed.

**** Note:** Historical research indicates that a 3 per cent packet loss results in a MOS score dropping by 0.5 (out of 5).

F-5.1 PAMS

The Perceptual Analysis/Masurement System (PAMS) was developed by British Telecom, and uses a mathematical model of human hearing. PAMS measurements report two scores: Listening Quality (LQ) and Listening Effort (LE).

These are both on a five-point scale similar to the MOS score.

**** Note:** Research shows that PAMS scores average within a half-point when compared to traditional MOS scores.

**** Note:** In PAMS measurements, errors in input signals are taken into consideration.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

F-5.2 PSQM and PSQM+

The Perceptual Speech Quality Measure (PSQM) was developed by the ITU (the International Telecommunications Union) as standard P.861. It is designed to measure the effect of compression on voice quality; it does not take lost packets into account.

PSQM scores are on a scale of 0 to 6.5, with 0 indicating no distortion and 6.5 indicating extreme distortion. (Thus, the lower the number, the better the voice quality.) There is no direct correlation between 'traditional' MOS scores and PSQM scores.

PSQM+ was developed to address some of these issues. PSQM+ performs additional post processing on PSQM values, and its results correlate more directly with 'traditional' MOS scores. It is to be noted, however, that PSQM+ still has issues with testing which involves lost packets.

F-5.3 PESQ

The Perceptual Evaluation of Speech Quality (PESQ) is presently the most advanced method for voice quality measurement. It was designed to succeed PSQM, and is defined in ITU standard P.862.

PESQ combines the techniques of PSQM+ and PAMS, and takes into account distortion, errors, packet loss, and delay.

PESQ measurement uses a sensory model (the comparison of the original ('reference') signal with the received ('degraded') signal), and its scores are analagous to MOS scores, with the best PESQ score which can be achieved being 4.5.

F-6 Unit and Integration Verification and Validation

The specific elements tested are:

- Two equivalent Dell Latitude D630 laptops as processing nodes (PC1 and PC2)
- Two GL Communications Universal Telephony Agents (UTAs) (UTA155155 and UTA155156)
- Connectors and cabling necessary for interconnection
- GL Communications VQuad (running on both PC1 and PC2)
- GL Communications VQT (running on either PC1 or PC2)

The tested configurations are:



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

- PC1-to-UTA155155 RTP Transfer
- PC1-to-UTA155156 RTP Transfer
- PC2-to-UTA155155 RTP Transfer
- PC2-to-UTA155156 RTP Transfer

Note: Allow all current transfer/recording processes to complete to avoid truncating a recorded file. Incomplete recordings will produce poor VQT scores.

The tests consist of three iterations ('runs') of each:

- 1 encoded voice file transferred 20 times
- 1 encoded voice file transferred 250 times
- 4 different encoded voice files transferred 10 times
- 4 encoded different voice files transferred 250 times
- 6 different encoded voice files transferred 10 times
- 6 different encoded voice files transferred 250 times

All transfers are bidirectional, with each DUT transmitting and receiving. The results of each run are tabulated and averaged for each Date/Time Group (DTG) of the run for the following data points:

1. PAMS_LE
2. PAMS_LQ
3. PSQM
4. PSQM_MOS
5. PSQMPlus
6. Calculated PSQMPlus_MOS
7. PESQ_Score
8. PESQ_LQ
9. PESQ_MOS



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

References

1. Department of Defense Unified Capabilities Requirements 2008 (UCR 2008)
2. Fusion *Voice Engine User's Manual*, Software Version 3.2.X, Part Number/Version: FVE V3.2.X, Release Date: November, 2010
3. GL Communications *Voice Quality Testing (VQT) User Guide*, Updated June 2010
4. GL Communications *VQuad™ (VOICE Quad) User Manual*, Updated May 2008
5. GL Communications *GL VQT Reference*, Log Output, September 2004
6. *Packetcable Implementation*, Jeff Riddel, Copyright 2007, Cisco Systems, Inc.



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

APPENDIX G References



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

References

iSTAT, In-Stat: VoIP Security: Preparing for the evolving threat, September 2006.

The Internet Society, Network Working Group, “SIP: Session Initiation Protocol”, Request For Comments 3261, June 2002. , <http://www.ietf.org/rfc/rfc3261.txt>, July 6, 2007.

The Internet Society, Network Working Group, “Communications Resource Priority for the Session Initiation Protocol (SIP)”, Request For Comments 4412, February 2006.

The Internet Society, Network Working Group, “Extending the Session Initiation Protocol (SIP) Reason Header for Preemption Events”, Request For Comments 4411, February 2006.

[Light Reading, VoIP Reliability, June 2004](http://www.lightreading.com/document.asp?doc_id=53864&page_number=4)

http://www.lightreading.com/document.asp?doc_id=53864&page_number=4.

Recommended IP Telephony Architecture, National Security Agency’s Systems and Network Attack Center (SNAC), <http://www.nsa.gov/snac/voip/I332-009R-2006.pdf>, September 12, 2007.

Developed by DISA for the DOD, “IPT & VoIP STIG, V2R2 21 April 2006, Internet Protocol Telephony & Voice Over Internet Protocol, Security Technical Implementation Guide V2R2”.

Developed by DISA for the DOD, “Network Infrastructure, Security Technical Implementation Guide V6R4.

Security Guidance for Deploying IP Telephony Systems, National Security Agency’s Systems and Network Attack Center (SNAC), <http://www.nsa.gov/snac/voip/I332-016R-2005.PDF>, September 12, 2007.

Das, K. P., Hubbart, J., and T. Rumland, “Intelligent Advanced Communications - Network Infrastructure,” April 2, 2007.

Recommendation H.323, International Telecommunications Union
<http://www.itu.int/rec/T-REC-H.323/e>, July 16, 2007.

Session Initiation Protocol - Wikipedia, the free encyclopedia,
http://en.wikipedia.org/wiki/Session_Initiation_Protocol, July 16, 2007.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

The Opportunity: Pervasive Computing - Indiana University
http://www.indiana.edu/~ovpit/ipcres/index_4.html, July 16, 2007.

The Internet Society, Network Working Group, “SIP: Session Initiation Protocol”, Request For Comments 2543, March 1999, (Obsolete). RFC 2543,
<http://www.ietf.org/rfc/rfc2543.txt>, July 6, 2007.

The Internet Society, Network Working Group, “**Models for Multi Party Conferencing in SIP**”, Internet Draft draft-rosenberg-sip-conferencing-models-00, November 2000.

The Internet Society, Network Working Group, “Basic Network Media Services with SIP”, Request For Comments 4240, December 2005.

The Internet Society, Network Working Group, “**An Approach to Call Park/Retrieve Using SIP**”, Internet Draft draft-procter-sipping-call-park-extension-00, April; 2004.

CyberData, Pager Server <http://www.cyberdata.net/products/voip/voip-pagingserver.html>, September 24, 2007.

Raytheon, ARA-1, <http://www.jps.com/page/view/213>, September 24, 2007.

K. S. Vallerio, L. Zhong, and N. K. Jhu, “Energy-Efficient Graphical User Interface Design”.

D. Kuhn, T. Walsh, and S. Fries, “Security Considerations for Voice Over IP Systems”, Recommendations of the National Institute of Standards and Technology, Special Publication 800-58, January 2005.

Dan York, CISSP “VoIP Security: How Secure is your IP Phone”.

<http://iase.disa.mil/stigs/index.html>, “Security Technical Implementation Guides (STIGS) and Supporting Documents” August 29, 2007.

Developed by DISA for the DOD, “Enclave Security Technical Implementation Guide,” Version 3, Release 2, July 28, 2005.

Developed by DISA for the DOD, “Defense Switched Network Security Technical Implementation Guide,” Version 2, Release 3, April 30, 2006.

Developed by DISA for the DOD, “UNIX Security Technical Implementation Guide,” Version 5, Release 1, March 28, 2006.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

D. Kuhn, T. Walsh, and S. Fries, “Security Considerations for Voice Over IP Systems”, Recommendations of the National Institute of Standards and Technology, Special Publication 800-58, January 2005.

Federal Information Processing Standards Publication, “Security Requirements for Cryptographic Modules”, FIPS 140-2, May 25, 2001.

International Standards Organization, “Information Technology–Security Techniques–Evaluation Criteria for IT Security, Part 1”, ISO/IEC 15408-1, 2nd edition, October 10, 2005.

Telcordia, “Generic Requirements for Network Element/Network System (NE/NS) Security”, GR-815-CORE Issue 2, March 2002.

Chairman of the Joint Chiefs of Staff Instruction, “Policy for Department of Defense Voice Networks”, CJCSI 6215.01B, September 23, 2001.

Chairman of the Joint Chiefs of Staff Instruction, “Information Assurance (IA) and Computer Network Defense (CND)”, CJCSI 6510.01D, June 15, 2004.

Chairman of the Joint Chiefs of Staff Instruction, “Defense Information System Network (DISN): Policy, Responsibilities and Processes”, CJCSI 6211.02B, July 31, 2003.

Ken Coar, “The open source definition,” July 24, 2006, June 27, 2007.
“<http://www.opensource.org/docs/definition.php>”

Eric S. Raymond, “The Cathedral and the Bazaar” “<http://www.free-software.org/literature/papers/esr/cathedral-bazaar/>”, September 25, 2007.

Blane Warrene, “Navigating Open Source Licensing,” March 9, 2005, June 27, 2007.
“<http://www.sitepoint.com/article/open-source-licensing>”

John Koenig, “Seven open source business strategies for competitive advantage,” May 13, 2004, June 27, 2007 “<http://www.itmanagersjournal.com/feature/314>”

LGS Innovations, LLC, “Intelligent Advanced Communications –VoIP Security Study”, May 4, 2007.

M. A. Padlipsky, “A Perspective on the ARPANET Reference Model,” RFC 871, September 1982.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

International Organization for Standards/International Electrotechnical Commission, ISO/IEC 7498-1, 7498-2, 7498-3, and 7498-4, “Open Systems Interconnection: Basic Reference Model,” available from www.iso.org .

International Organization for Standards/International Electrotechnical Commission, ISO/IEC 14496-10, “Coding of audio-visual objects – Part 10: Advanced Video Coding,” and 14496-10/FDAm1, “Support for color spaces and aspect ratio definitions,” available from www.iso.org.

International Telecommunication Union, ITU-T Recommendation H.261, “Video codec for audiovisual services at p x 64 kbit/s”, <http://www.itu.int/rec/T-REC-H.261-199303-I/en> .

International Telecommunication Union, ITU-T Recommendation H.263, “Video coding for low bit rate communication”, <http://www.itu.int/rec/T-REC-H.263-200501-I/en> .

International Telecommunication Union, ITU-T Recommendation H.264, “Advanced video coding for generic audiovisual services”, <http://www.itu.int/rec/T-REC-H.264-200503-I/en>.

International Telecommunication Union, ITU-T Recommendation H.264 (2005) Amendment 1 (06/06), “Support of additional color spaces and removal of the High 4:4:4 Profile,” <http://www.itu.int/rec/T-REC-H.264-200606-I!Amd1/en>.

International Telecommunication Union, ITU-T Recommendation G.711, “Pulse code modulation (PCM) of voice frequencies”, <http://www.itu.int/rec/T-REC-G.711/en> .

International Telecommunication Union, ITU-T Recommendation G.726, “40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)”, <http://www.itu.int/rec/T-REC-G.726/en>.

International Telecommunication Union, ITU-T Recommendation G.722, “7KHz audio-coding within 64 kbit/s”, <http://www.itu.int/rec/T-REC-G.722/en>.

International Telecommunication Union, ITU-T Recommendation G.722.1, “Low-complexity coding at 24 and 32 kbit/s for hands-free operation in systems with low frame loss”, <http://www.itu.int/rec/T-REC-G.722.1/en> .

International Telecommunication Union, ITU-T Recommendation G.722.2, “Wideband coding of speech at around 16 kbit/s using Adaptive Multi-Rate Wideband (AMR-WB),” <http://www.itu.int/rec/T-REC-G.722.2/en>.

International Telecommunication Union, ITU-T Recommendation G.728, “Coding of speech at 16 kbit/s using low-delay code excited linear prediction”, <http://www.itu.int/rec/T-REC-G.728/en> .



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

International Telecommunication Union, ITU-T Recommendation G.723.1, “Dual rate speech coder for multimedia communication transmitting at 5.3 and 6.3 kbit/s”, <http://www.itu.int/rec/T-REC-G.723.1-200605-P/en>.

International Telecommunication Union, ITU-T Recommendation G.729, “Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP),” (Status Pre-published), January 2007, <http://www.itu.int/rec/T-REC-G.729/en>.

International Telecommunication Union, ITU-T Recommendations G.729 Annex A, “Reduced Complexity 8 kbit/s CS-ACELP speech codec”, (Status- superseded), <http://www.itu.int/rec/T-REC-G.729-199611-S!AnnA/en>.

International Telecommunication Union, ITU-T Recommendations G.729 Annex B, “A silence compression scheme for G.729 optimized for terminals conforming to Recommendation V.70”, (Status- superseded), <http://www.itu.int/rec/T-REC-G.729-199610-S!AnnB/en>.

International Telecommunication Union, ITU-T Recommendation G.729.1, “G.729.1 : G.729 based Embedded Variable bit-rate coder: An 8-32 kbit/s scalable wideband coder bitstream interoperable with G.729,” (Status - In force), May 2006, <http://www.itu.int/rec/T-REC-G.729.1/en>

Herlein, G., Morlat, S., Jean-Marc, J., Hardiman, R., and P. Kerr, “RTP Payload Format for the Speex Codec,” draft-herlein-speex-rtp-profile, April 4, 2005.

Andersen, S., Duric, A., Astrom, H., Hagen, R., Kleijn, W., and J. Linden, “Internet Low Bit Rate Codec (iLBC),” RFC 3951, December 2004.

Jennings, C., Peterson, J., and M. Watson, “Private Extensions to the Session Initiation protocol (SIP) for asserted Identity within Trusted Networks”, RFC 3325, November 2002.

VoIPForo.com, http://www.en.voipforo.com/H323/H323_example.php September 25, 2007.

VoIPForo.com, http://www.en.voipforo.com/SIP/SIP_example.php September 25, 2007.

VoIPForo.com, <http://www.en.voipforo.com/H323vsSIP.php> September 25, 2007.

Freshmeat.net : About, <http://freshmeat.net/about/>, July 6, 2007.

Asterisk: The Open Source Telephony Platform | About, <http://www.asterisk.or/about>, July 2, 2007.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Wiki CallWeaver, <http://www.callweaver.org/wiki/CallWeaver>, July 5, 2007.

FreeSwitch – Communication Consolidation, <http://www.freeswitch.org/>, July 5, 2007.

Comparing sipX with Asterisk – SIPfoundary sipx, The Open Source SIP PBX for linux – Calvia, http://sipx-wiki.calivia.com/index.php/Comparing_sipX_with_Asterisk, July 5, 2007.

Yate – Main – Homepage, <http://yate.null.ro/pmwiki/>, July 5, 2007.

Ekiga ~ Free Your Speech, <http://www.ekiga.org>, July 2, 2007.

Kphone, <http://sourceforge.net/projects/kphone/>, July 2, 2007.

Minisip, <http://www.minisip.org>, July 2, 2007.

Zap! – The Mozilla SIP Client, <http://www.croczilla.com/zap>, July 5, 2007.

The eXtended oslibrary, <http://savannah.nongnu.org/projects/exosip>, July 2, 2007.

ain-sip, Java API for SIP Signalling, <https://jain-sip.dev.java.net/>, July 5, 2007.

OpenSourceSIP: OSS – Open Source SIP,
<http://www.opensourcesip.org:8080/jiveforums/opensipstack.jsp>, July 2, 2007.

The GNU oSIP Library, <http://www.gnu.org/software/osip>, July 2, 2007.

Vox Gratia FAQ, <http://www.voxgratia.org/docs/faq.html>, July 5, 2007.

PJSIP – Open Source Embedded SIP Stack and Media Stack, July 5, 2007.

Resip Overview – Resiprocate, http://www.resiprocate.org/Resip_Overview, July 5, 2007.

Sofia-SIP Library, <http://sofia-sip.sourceforge.net/>, July 3, 2007.

OpenSourceSIP: OSS – Open Source SIP,
<http://www.opensourcesip.org:8080/jiveforums/downloads.jsp>, July 2, 2007.

Mobicents.org – The Open Source VoIP Middleware Platform,
<http://www.mobicents.org>, July 2, 2007.

Home – Open Source SIP Server, <http://www.openser.org/>, July 5, 2007.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

About Sip Express Router, <http://www.iptel.org/ser/>, July 5, 2007.

J.G. Andrews, A. Ghosh, and R. Muhamed, *Fundamentals of WiMAX: Understanding Broadband Wireless Networking*, Prentice Hall, 2007.

G.J. Foschini and M.J. Gans, "On Limits of Wireless Communications in a Fading Environment when Using Multiple Antennas," *Wireless Personal Communications*, Kluwer Academic Publishers, Vol. 6, No. 3, pp 311–335, March 1998.

A. Goldsmith, S. Ali Jafar, H. Jindal, and S. Vishwanath, "Fundamental Capacity of MIMO Channels," *IEEE Journal on Selected Areas in Communication*, Vol. 21, No. 5, pp 684–702, June 2003.

J. Winters, "Understanding MIMO,"
<http://www.wirelessnetdesignline.com/showArticle.jhtml?articleID=161500272>.

J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley and E. Schooler, "SIP: Session Initiation Protocol," IETF RFC 3261, June 2002.

N. Borisov, I. Goldberg, D. Wagner, "Intercepting mobile communications: the insecurity of 802.11," *7th Annual international Conference on Mobile Computing and Networking–MobiCom '01*, pp 180-189, 2001.

S. Fluhrer, I. Mantin, A. Shamir, "Weaknesses in the Key Scheduling Algorithm of RC4," *Selected Areas in Cryptography: 8th Annual International Workshop*, pp 1–24, August 2001.

J.R. Walker, "Unsafe at any key size; an analysis of the WEP encapsulation," IEEE Document 802.11-00/362, October 2000.

J.R. Rao, P. Rohatgi, H. Scherzer and S. Tinguely, "Partitioning Attacks: Or how to Rapidly Clone Some GSM Cards," *IEEE Symposium on Security and Privacy*, May 2002.

D. Wagner, I. Goldberg, M. Briceno, "GSM Cloning," April 1998,
<http://www.isaac.cs.berkeley.edu/isaac/gsm-faq.html>.

E. Barkan, E. Biham, and B. Keller, "Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication," *Advances in Cryptology–CRYPTO 2003*, vol. 2729/2003, pp. 600–616, October 2003.

C. Wingert and M. Naidu, "CDMA 1xRTT Security Overview," Qualcomm Whitepaper, August 2002.

News Release, "Sprint Nextel Announces 4G Wireless Broadband Initiative with Intel, Motorola and Samsung," August 8, 2006.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

K. S. Vallerio, L. Zhong, and N. K. Jhu, “Energy-Efficient Graphical User Interface Design”.

J. Polk, “Extending the Session Initiation Protocol (SIP) Reason Header for Preemption Events RFC 4411”.

H. Schulzrinne, and J. Polk, “Communications Resource Priority for the Session Initiation Protocol (SIP) RFC 4412.

Cisco Systems “Understanding, Preventing and Defending Against Layer 2 Attacks Session SEC-2002 <http://www.cisco.com/networkers/nw04/presos/docs/SEC-2002.pdf>.

F. Robles. “The VoIP Dilemma”, SANS Institute,
<http://www.sans.org/rr/whitepapers/voip/1452.php>.

VQIPSA, “VoIP Security and Privacy Threat Taxonomy”.

R. Rivest, A. Shamir, L. Adleman, “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems,” Communications of the ACM, Vol. 21, No. 2, pp. 120–126, February 1978.

Federal Information Processing Standards Publication, “Digital Signature Standard (DSS),” FIPS 186, May 19, 1994.

R. Rivest, “The MD5 Message-Digest Algorithm,” IETF RFC 1321, April 1992.

Federal Information Processing Standards Publication, “Secure Hash Standard,” FIPS 180-2, April 17, 1995.

Federal Information Processing Standards Publication, “Data Encryption Standard (DES),” FIPS 46-3, October 25, 1999.

Federal Information Processing Standards Publication, “Advanced Encryption Standard (AES),” FIPS 197, November 26, 2001.

International Telecommunication Union (ITU-T) X.509 “Information technology – Open Systems Interconnection - The Directory: Public-Key and Attribute Certificate Frameworks,” August 2005 (revised).

DoD Instruction, “Public Key Infrastructure (PKI) and Public Key (PK) Enabling,” DoDI 8520.2, April 1, 2004.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

DoD, “X.509 Certificate Policy for the United States Department of Defense,” Version 9.0, February 9, 2005.

T. Dierks and C. Allen, “The TLS Protocol Version 1.0,” IETF RFC 2246, January 1999.

E. Rescorla, “HTTP Over TLS,” IETF RFC 2818, May 2000.

R. Housley, W. Ford, W. Polk and D. Solo, “Internet X.509 Public Key Infrastructure Certificate and CRL Profile,” IETF RFC 2459, January 1999.

M. Baugher, D. McGrew, M. Naslund, E. Carrara and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” IETF RFC 3711, March 2004.

M. Euchner, “HMAC-Authenticated Diffie-Hellman for Multimedia Internet KEYing (MIKEY),” IETF RFC 4650, September 2006.

J. Arkko, E. Carrara, F. Lindholm, M. Naslund and K. Norrman, “MIKEY: Multimedia Internet KEYing,” IETF RFC 3830, August 2004.

S. Kent, K. Seo, “Security Architecture for the Internet Protocol,” IETF RFC 4301, December 2005.

S. Kent, “IP Authentication Header,” IETF RFC 4302, December 2005.

S. Kent, “IP Encapsulating Security Payload (ESP),” IETF RFC 4303, December 2005.

J. Schiller, “Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2),” IETF RFC 4307, December 2005.

C. Kaufman, “Internet Key Exchange (IKEv2) Protocol,” IETF RFC 4306, December 2005.

H. Orman, “The OAKLEY Key Determination Protocol,” IETF RFC 2412, November 1998.

W. Diffie and M. Hellman, “New Directions in Cryptography,” IEEE Transactions on Information Theory, Vol. 22, No. 6, pp. 644–654, November 1976.

D. Harkins and D. Carrel, “The Internet Key Exchange (IKE),” IETF RFC 2409, November 1998.

F. Andreasen, M. Baugher and D. Wing, “Session Description Protocol (SDP) Security Descriptions for Media Streams,” IETF RFC 4568, July 2006.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

Van de Velde, G., Hain, T., Droms, R., Carpenter, B., and E. Klein, “Ipv6 Network Architecture Protection,” draft-vandeveld-v6ops-nap-01, January 24, 2005.

W. Eddy, “Comparison of IPv4 and IPv6 Header Overhead,” draft-eddy-ipv6-overhead-00, May 8, 2006.

M. Baugher, D. McGrew, M. Naslund, E. Carrara and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” IETF RFC 3711, March 2004.

F. Andreassen, M. Baugher and D. Wing, “Session Description Protocol (SDP) Security Descriptions for Media Streams,” IETF RFC 4568, July 2006.

J. Arkko, E. Carrara, F. Lindholm, M. Naslund and K. Norrman, “MIKEY: Multimedia Internet KEYing,” IETF RFC 3830, August 2004.

DISA, “Department of Defense Voice Networks Generic Switching Center Requirements (GSCR),” September 8, 2003 (Errata Change 2, December 14, 2006).



Maritime Systems

**Intelligent Advanced Communications IP Telephony
Feasibility for the US Navy – Phase 3**

This page intentionally left blank.



Maritime Systems

Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Symbols, Abbreviations, and Acronyms



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

3DES	Triple DES
3GPP	3rd Generation Partnership Project
3GPP2	3rd Generation Partnership Project 2
3PCC	Third Party Call Control
AAP	Alarm Activation Panel
AC	Access Categories
ACF	Admission Confirm message
ACL	Access Control List
ADC	Analog-To-Digital Converter
ADM	Administrator
ADNS	Automatic Digital Network System
AES	Advanced Encryption Standard
AFE	Analog Front End
AH	Authentication Header
AMR	Adaptive Multi-Rate
AP	Audio Panel
API	Application Programming Interface
ARP	Address Resolution Protocol
ARPA	Advanced Research Project Agency
ARPANET	Advanced Research Projects Agency Network
ARQ	Admission Request message
ASCII	American Standard Code for Information Interchange
ASLAN	Assured Services Local Area Network
ASN	Abstract Syntax Notation
AS-SIP	Assured Services-Session Initiation Protocol
ASM	Asynchronous Transfer Mode



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

BISDN	Broadband Integrated Services Digital Network
B2BUA	Back-To-Back User Agent
BPSK	Binary Phased Shift Keying
BRI	Basic Rate Interchange
BSD	A family of permissive free software licenses. The original was used for the Berkeley Software Distribution.
BSR	Base Station Router
CA	Certificate Authority
CAAS	Central Amplifier Announcing System
CAG	Carrier Air Group
CALEA	Communications Assistance for Law Enforcement Act
CAM	Content Addressable Memory
CANES	Consolidated Afloat Networks and Enterprise Services
CAS	Common Associated Signaling
CC	Common Criteria
CD	Compact Disc
CDMA	Code Division Media Access
CDP	Cisco Discovery Protocol
CDR	Call Detail Record
CEM	Customer Experience Management
CER	Customer Edge Router
CISSP	Certified Information Systems Security Professional
CJCS	Chairman of the Joint Chiefs of Staff
CJCSI	Chairman of the Joint Chiefs of Staff Instruction
CLC	Close Logical Channel
CNG	Comfort Noise Generator
CODEC	Coder/Decoder
CoS	Class of Service



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

COTS	Commercial-Off-The-Shelf
CPE	Customer Premises Equipment
CPIM	Common Presence and Instant Messaging
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CRL	Certificate Revocation List
CSMA	Carrier Sensed Multiple Access
CT	Connecticut
CVN	Carrier Vessel Nuclear
DAC	Digital-To-Analog Converter
DARPA	Defense Advanced Research Projects Agency
DARTS	DISN Assured RTS Support
DCF	Disengage Confirm
DDG	Guided Missile Destroyer
DDRE	Dual-Decision Feedback Equalizer
DES	Data Encryption Standard
DH	Diffie-Hellman
DHCP	Dynamic Host Configuration Protocol
DID	Direct Inward Dial
DISA	Defense Information System Agency
DNS	Domain Name Service
DOCSIS	Data Over Cable Service Interface Specification
DoD	Department of Defense
DoDI	DOD Instruction
DoS	Denial of Service
DRM	Digital Rights Management
DRQ	Disengage Request
DRSN	Defense Red Switch Network



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

DS	Dedicated Station
DSA	Digital Signature Algorithm
DSN	Defense Switched Network
DSP	Digital Signal Processing
DSSS	Direct Sequence Spread Spectrum
DT	Dial Telephone
DTMF	Dual-Tone Multi-Frequency
DTP	Dynamic Trunking Protocol
DUT	Device Under Test
DVD	Digital Versatile Disc
DVMRP	Distance Vector Multicast Routing Protocol
DVX	Deployable Voice Exchange
EAL	Evaluated Assurance Level
EBC	Edge Boundary Controller
E&M	Ear and Mouth
EAP	Extensible Authentication Protocol
EAP-TLS	Extensible Authentication Protocol -Transport Layer Security
EAP-TTLS	Extensible Authentication Protocol -Tunneled Transport Layer Security
EAPOL	Extensible Authentication Protocol Over LAN
EBCDIC	Extended Binary Coded Decimal Interchange Code
ECAS	Electronic Call Accounting System
ECS	Exterior Communications System
EI	End Instrument
EIA	Electronic Industries Association
E&M	Ear and Mouth
EMCON	Emission Controls
EO	End Office
EPCC	End-Point Call Control



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

ESC	End Session Command
ESP	Encapsulated Security Payload
FDM	Frequency Division Multiplexing
FEC	Forward Error Correction
FFT	Fast Fourier Transform
FHSS	Frequency Hopping Spread Spectrum
FIPS	Federal Information Processing Standards
FSB	Fleet Select Box
FTP	File Transfer Protocol
FXS	Foreign Exchange Station
GA	General Alarm
GARP	Gratuitous Address Resolution Protocol
GCCS	Global Command and Control System
GIG	Global Information Grid
GigE	Gigabit Ethernet
GIPS	Global IP Solutions Inc.
GMSK	Gaussian Minimum Shift Keying
GNU	Recursive acronym for "GNU's Not Unix!"
GPHY	Gigabit Physical
GPL	General Public License
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile
GSTN	General Switched Telephone Network
GUI	Graphical User Interface
HA	High Availability
HAIPE	High Assurance Internet Protocol Encryptor
HMAC	Hashed Message Authentication Code



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

HPI	Host Port Interface
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Over Secure Socket Layer
IA	Information Assurance
iAC	Intelligent Advanced Communications
iACT	Intelligent Advanced Communications Terminal
IAW	In accordance with
IANA	Internet Assigned Numbers Authority
IAX	Inter-Asterisk Exchange Protocol
IC	Integrated Communication
ICAN	Integrated Communications and Advanced Networks
ICMP	Internet Control Message Protocol
ICMPv6	Internet Control Message Protocol for IPv6
ICS	Integrated Communications System
ICSAP	Integrated Communications System Audio Panel
ICSCU	ICS Control Unit
ICT	Integrated Communication Terminal
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IKE	Internet Key Exchange
iLBC	internet Low Bitrate Codec
IM	Instant Messaging
IOM	ISDN-Oriented Modular
IP	Internet Protocol; Inter-Phone
IPDC	IP Device Control
IP-PBX	IP-Public Branch Exchange



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

IPSec	IP Security
IPV4	Internet Protocol Version 4
IPV6	Internet Protocol Version 6
IS	Intrinsically Safe
ISAKMP	Internet Security Association and Key Management Protocol
ISDN	Integrated Services Digital Network
ISO	International Standards Organization
ITU	International Telecommunication Union
ITU-T	ITU Telecommunication Standardization Sector
IVCN	Integrated Voice Communications Network
IVN	Integrated Voice Network
IVUT	Integrated Voice User Terminal
JITC	Joint Interoperability Test Command
JTAG	Joint Test Action Group
LAN	Local Area Network
LATA	Local Access and Transport
LCD	Liquid Crystal Display
LCS	Littoral Combat Ship
LEAP	Lightweight Extensible Authentication Protocol
LGS	LGS Innovations, LLC, Subsidiary of Alcatel-Lucent
LGPL	Lesser General Public License
LLC	Logical Link Control; Limited Liability Company
LMR	Land Mobile Radio
LSC	Local Session Controller
LSP	Label-Switch Path
LSSGR	Local Switching Systems Generic Requirements
LTE	Long Term Evolution
LTIU	Lockout Trunk Interface Unit



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

MAC	Media Access Control
Mbps	Megabit Per Second
MC	Multi-Channel
MCU	Multipoint Control Units
MFS	Multifunction Switch
MG	Media Gateway
MGCP	Media Gateway Control Protocol
MIB	Management Information Base
MIKEY	Multimedia Internet Keying
MIME	Multipurpose Internet Mail Extensions
MIMO	Multiple-Input/Multiple-Output
MIPS	Microprocessor without Interlocked Pipeline Stages
MitM	Man in the Middle Attacks
MLPP	Multi-Level Precedence and Preemption
MMU	Memory Management Unit
MMUSIC	Multiparty Multimedia Session Control
MOS	Mean Opinion Score
MPLS	Multi Protocol Label Switching
MSTP	Multiple Spanning Tree Protocol
MWS	MORIAH Wind System
NAPT	Network Address Port Translation
NAT	Network Address Translators
NAVSSI	Navigation Sensor System Interface
NIC	Network Interface Card
NIPRNet	Non-Classified Internet Protocol Router Network
NIST	National Institute of Standards and Technology
NLOS	Near Line of Sight
NMS	Network Management System



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

NSA	National Security Agency
NVP	Network Voice Protocol
OAMP	Operations, Administration, Maintenance, And Provisioning
OCSP	Online Certificate Status Protocol
OEM	Original Equipment Manufacturer
OFDM	Orthogonal Frequency Division Multiplexing
OFDMA	Orthogonal Frequency Division Media Access
OLC	Open Logical Channel
OMA	Open Mobile Alliance
ONR	Office of Naval Research
OOB	Out-Of-Band
OOP	Object Oriented Programming
OS	Operating System
OSI	Open System Interconnection
OSPF	Open Shortest Path First
PA	Public Address; Presence Agents
PAGP	Port Aggregation Protocol
PAMS	Perceptual Analysis/Measurement System
PAMS_LE	Perceptual Analysis/Measurement System Listening Effort
PAMS_LQ	Perceptual Analysis/Measurement System Listening Quality
PAN	Personal Area Network
PBX	Private Branch Exchange
PBX1	Private Branch Exchange Type 1
PBX2	Private Branch Exchange Type 2
PC	Personal Computer
PCAP	Packet Capture
PCM	Pulse-Code Modulation



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

PDA	Personal Digital Assistant
PER	Pack Encoding Rules
PESQ	Perceptual Evaluation of Speech Quality
PESQ_LQ	Perceptual Evaluation of Speech Quality Listening Quality
PESQ_MOS	Perceptual Evaluation of Speech Quality Mean Opinion Score
PESQM	Perceptual Speech Quality Measure
PESQM_MOS	Perceptual Speech Quality Measure Mean Opinion Score
PHY	Generic electronics term referring to a special electronic integrated circuit or functional block of a circuit that takes care of encoding and decoding between a pure digital domain (on-off) and a modulation in the analog domain.
PICT	Programmable Integrated Communication Terminal
PKI	Public Key Infrastructures
PoC	Proof-of-Concept
PoE	Power Over Ethernet
POT	Plain Old Telephone
PP	Protection Profile
PRACK	Provisional Response Acknowledgement
PRI	Primary Rate Interface
PSTN	Public Switched Telephone Network
PTT	Push-To-Talk
PUA	Presence User Agent
PVST	Per VLAN Spanning Tree
QAM-16	Quadrature Amplitude Modulation-16
QAM-64	Quadrature Amplitude Modulation-64
QoS	Quality of Service
QPSK	Quadrature Phased Shift Keying
R&D	Research and Development
RADIUS	Remote Authentication Dial In User Service



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

RAID	Redundant Array of Independent Drives
RAM	Random Access Memory
RAS	Register, Admission, Status protocol
RCCS	Ruggedized Command & Control Solutions
RCS	Radio Communication System
RF	Radio Frequency
RFC	Request For Comments
RIP	Routing Information Protocol
RISC	Reduced Instruction Set Computer
RMII	Reduced Media Independent Interface
RPR	Resilient Packet Ring
RSA	Rivest, Shamir, Adleman
RSSE	Reduced-State Sequence Estimator
RSTP	Rapid Spanning Tree Protocol
RSVP	Resource Reservation Protocol
RTCP	Real-Time Control Protocol
RTD	Round-Trip Delay
RTOS	Real Time Operating System
RTP	Real-Time Transport Protocol
RTS	Real Time Services
SA	Security Association
SATCC	Ship Air Traffic Control Communications
SBC	Single Board Computer; Session Border Controller
SBU	Secure But Unclassified
SCCP	Skinny Client Control Protocol
SCD	Source Control Drawing
SCDS	Ships Control Display System
SCN	Switch Circuit Network



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

SCSI	Small Computer System Interface
SDK	Software Development Kits
SDP	Session Description Protocol
SDRAM	Synchronous Dynamic Random Access Memory
SES	SIP Enablement Server - Avaya
SGCP	Simple Gateway Control Protocol
SHA	Secure Hash Algorithm
SIP	Session Initiation Protocol
SIPRNet	Secret Internet Protocol Router Network
SISO	Single-Input/Single-Output
SMEO	Small End Office
S/MIME	Secure / Multipurpose Internet Mail Extension
SMTP	Simple Mail Transfer Protocol
SNAC	Systems and Network Attack Center
SNMP	Simple Network Management Protocol
SOAP	Simple Object Access Protocol; Service Oriented Architecture Protocol
SPAN	Switched Port Analyzer
SPAWAR	Space and Naval Warfare Systems Command
SPI	Serial Peripheral Interface Bus
SPIT	Spam Over Internet Telephony
SPT	Sound Powered Telephone; Sound Powered Telephony
SRTP	Secure Real-time Transport Protocol
SSL	Secure Socket Layer
SS7	System Signaling #7
SSN	Fast Attack Submarine, Ship Submersible, Nuclear
SSL	Secure Sockets Layer
STE	Secure Telephone Equipment; Secure Terminal Equipment
STIG	Security Technical Implementation Guide



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

STP	Spanning Tree Protocol
STU	Secure Terminal Units
STUN	Simple Traversal of UDP Through NATs
SVP	SpectraLink Voice Priority
T1	Digital signal 1 (also known as DS1) is a T-carrier signaling scheme widely used in telecommunications in North America and Japan to transmit voice and data between devices.
TCP	Transmission Control Protocol
TCS	Terminal Capability Set
TDM	Time-Division Multiplexing
TDMA	Time Division Multiple Access
TFTP	Trivial File Transfer Protocol
TI	Texas Instruments
TIA	Telecommunications Industry Association
TLB	Translation Lookaside Buffer
TLS	Transport Layer Security
ToS	Type of Service
TRP	Transport, Routing and Package
TSCE	Total Ship Computing Environment
TTL	Time to Live
TVS	Tactical Variant Switch
Tx/Rx	Transmit/Receive
UA	User Agent; Universal Alcatel
UAC	User Agent Client
UART	Universal Asynchronous Receiver/Transmitter
UAS	User Agent Server
UCR	Unified Capabilities Requirements 2008
UDDI	Universal Description, Discovery and Integration
UDLD	Uni-Directional Link Detection



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

UDP	User Datagram Protocol
UMB	Ultra Mobile Broadband
UMTS	Universal Mobile Telecommunication System
UNISTIM	Unified Networks IP Stimulus
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
US	United States
USB	Universal Serial Bus
UWB	Ultra-Wide Band
VLAN	Virtual Local Area Networks
VLYNQ	Proprietary interface developed by Texas Instruments
VM	Voice Mail
VMPS	VLAN Management Policy Server
VoIP	Voice Over Internet Protocol
VOMIT	Voice Over Misconfigured Internet Telephones
VOX	Voice Operated Switch
VPN	Virtual Private Network
VQP	VMPS Query Protocol
VRRP	Virtual Router Redundancy Protocol
VTP	VLAN Trunking Protocol
VWCS	Virtually Wirefree Communications System
VXML	Voice eXtensible Markup Language
WAN	Wide Area Network
WEP	Wired Equivalent Privacy or Wireless Encryption Protocol
WIFCOM	Wire Free Communication
Wi-Fi	“Wireless Fidelity”. Generically used to describe wireless interface of mobile computing devices, such as laptops in LANs . Wi-Fi® and the Wi-Fi CERTIFIED™ logo are registered trademarks of the Wi-Fi Alliance®.



Intelligent Advanced Communications IP Telephony Feasibility for the US Navy – Phase 3

Maritime Systems

WiMAX	Worldwide Interoperability for Microwave Access
WLAN	Wireless LAN
WME	Wireless Multimedia Extensions
WMM	Wi-Fi Multimedia
WPA/ WPA2	Wi-Fi Protected Access
WSDL	Web Services Description Language
XML	Extensible Markup Language